

UNIVERSIDAD DE EL SALVADOR

FACULTAD DE INGENIERIA Y ARQUITECTURA

ESCUELA DE INGENIERIA DE SISTEMAS INFORMATICOS

PROGRAMACION PARA DISPOSITIVOS MOVILES

PDM115 **Ciclo I-2014**

GUIA DE LABORATORIO N° 9 A

(Servicio Web Netbeans-java,Glassfish MySQL)

Objetivo:

Crear servicios web utilizando la arquitectura REST, para obtener información desde un servidor web montado localmente con el que se conecta una base de datos MySQL.

Requerimientos previos:

- Instalar WampServer(Version 2.2)
- Instalar Netbeans 7.3
- Conocer comandos DDL y DML en MySql.
- Instalar el addon “RESTClient, a debugger for RESTful web” para Mozilla Firefox

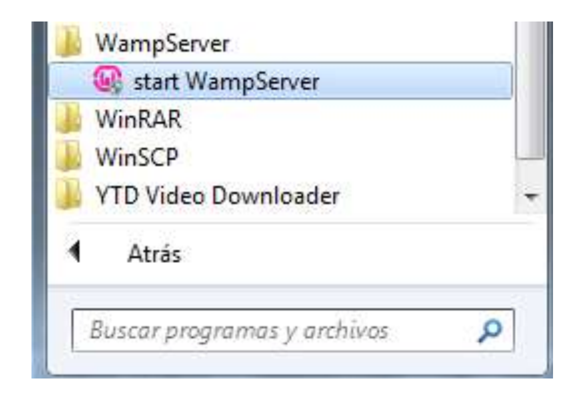
Indice

Tabla de contenido

Creación de Base de Datos

Primero iniciamos el wampserver

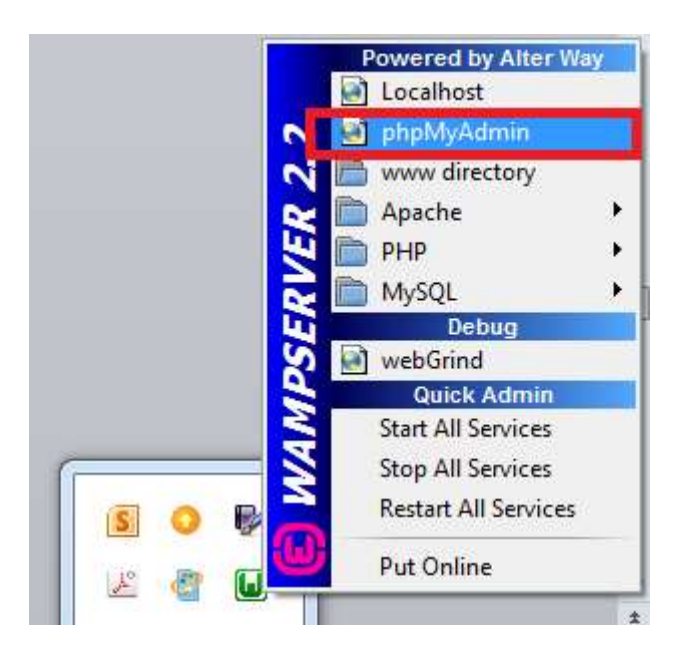
Clic en inicio, todos los programas, WampServer, start wampserver



Vea que este el icono en verde, para proseguir, si ve otro color, consulte a su instructor.

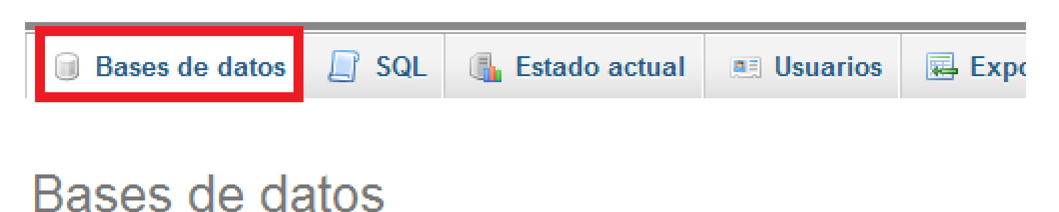


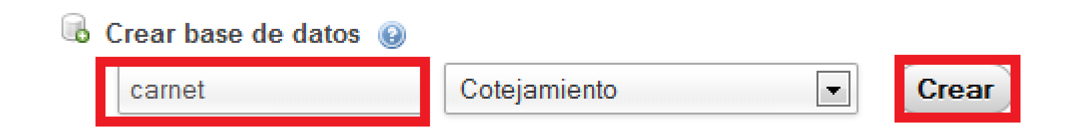
Presione el icono verde y entre a Phpmyadmin



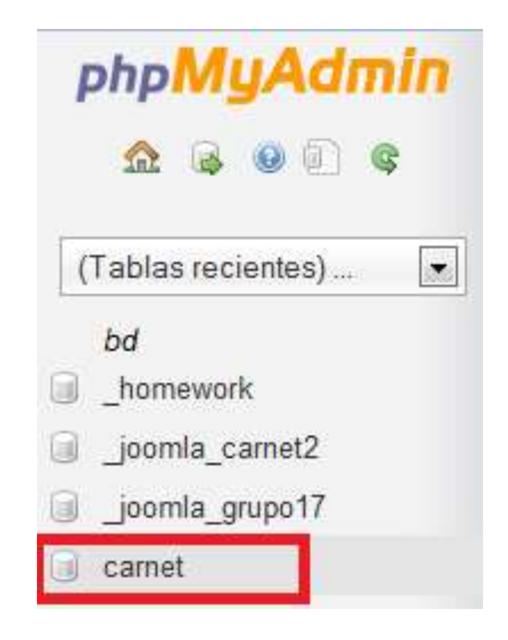
Crear la Base de Datos

Entre a base de datos, póngale nombre a su base como el de su carnet, luego presione crear.



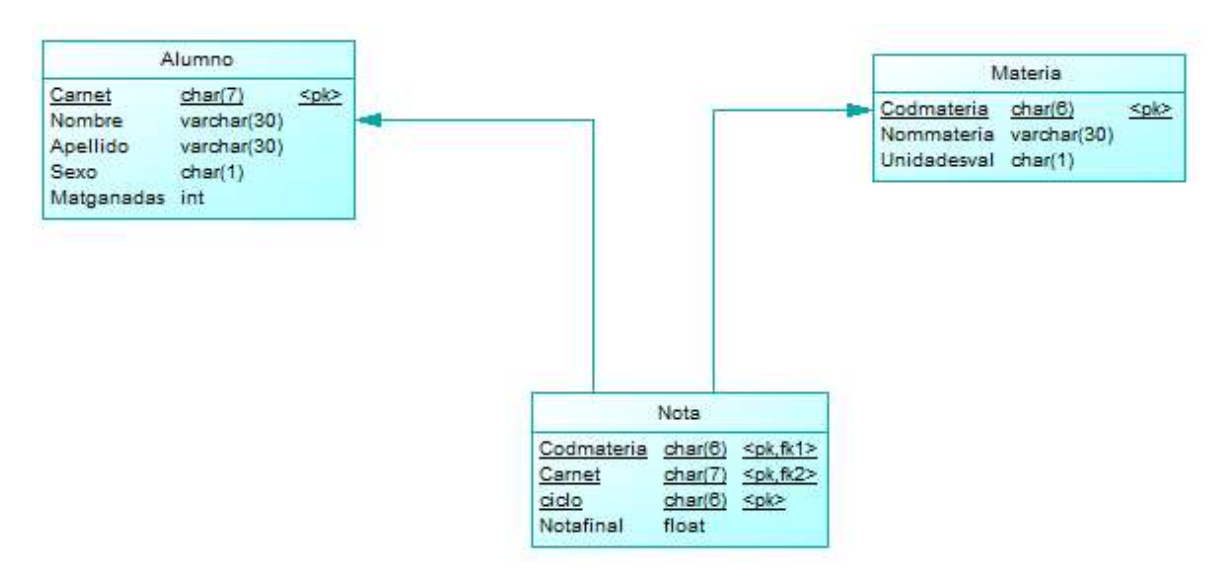


Después buscamos al lado izquierdo la base y presionamos clic sobre ella



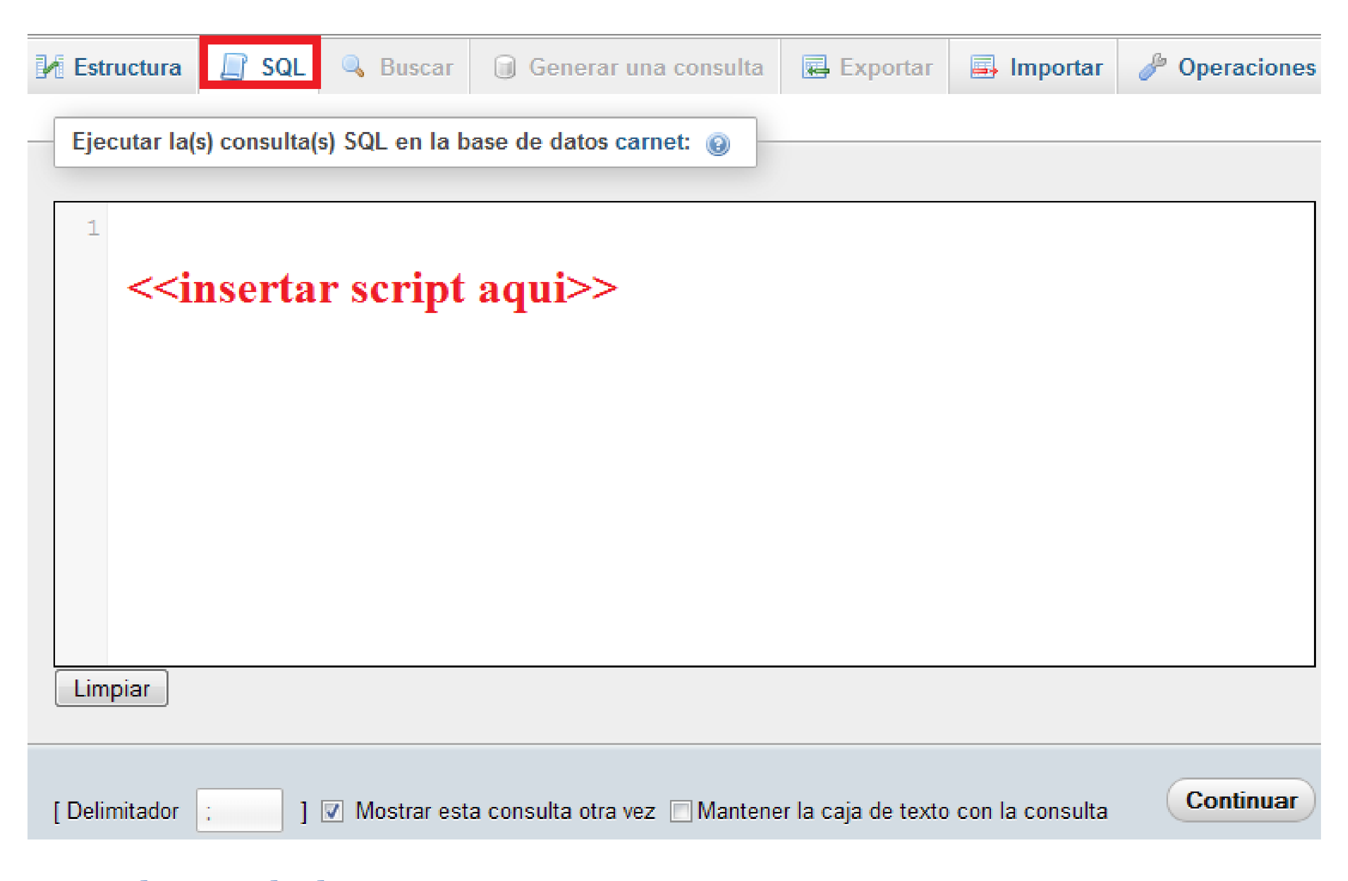
*Parte Opcional

Si deseamos crear la base de datos a partir de un modelo conceptual podemos utilizar el archivo **modelo_guia08.cdm y generar archivo físico(MySQL) y crear Script de Base de Datos.**



Fin de parte opcional

Presionamos clic en SQL e insertamos el script(abajo) en la parte que se indica



Script de Base de datos CARNET

```
/*==============================================================*/
/* DBMS name:      MySQL 5.0                                    */
/* Created on:     31/05/2013 18:04:40                          */
/*==============================================================*/
drop table if exists ALUMNO;
drop table if exists MATERIA;
drop table if exists NOTA;
/*==============================================================*/
/* Table: ALUMNO                                                */
/*==============================================================*/
create table ALUMNO
(
   CARNET               char(7) not null,
   NOMBRE               varchar(30) not null,
   APELLIDO             varchar(30) not null,
   SEXO                 char(1) not null,
   MATGANADAS           int,
```

```
  primary key (CARNET)
);
/*==============================================================*/
/* Table: MATERIA                                               */
/*==============================================================*/
create table MATERIA
(
  CODMATERIA           char(6) not null,
  NOMMATERIA           varchar(30) not null,
  UNIDADESVAL          char(1) not null,
  primary key (CODMATERIA)
);
/*==============================================================*/
/* Table: NOTA                                                  */
/*==============================================================*/
create table NOTA
(
  CODMATERIA           char(6) not null,
  CARNET               char(7) not null,
  CICLO                char(6) not null,
  NOTAFINAL            float,
  primary key (CODMATERIA, CARNET, CICLO)
);
alter table NOTA add constraint FK_POSEE foreign key (CODMATERIA)
    references MATERIA (CODMATERIA) on delete restrict on update restrict;
alter table NOTA add constraint FK_TIENE foreign key (CARNET)
    references ALUMNO (CARNET) on delete restrict on update restrict;
```

Luego presionamos continuar

Ahora actualice la página (F5 o clic derecho recargar) para ver sus tablas.



Prueba de llaves primarias:

(Probamos una por una la ejecución de las líneas de comando en cada tabla(SQL). La primera línea se debe ejecutar correctamente y la segunda debe generar error)

En Alumno

INSERT INTO ALUMNO VALUES ('NN00001','Nuevo','Nuevo','M',0);

INSERT INTO ALUMNO VALUES ('NN00001','Nuevo','Nuevo','M',0);

En Materia

INSERT INTO MATERIA VALUES('MAT115','MATEMATICAS I',4);

INSERT INTO MATERIA VALUES('MAT115','MATEMATICAS I',4);

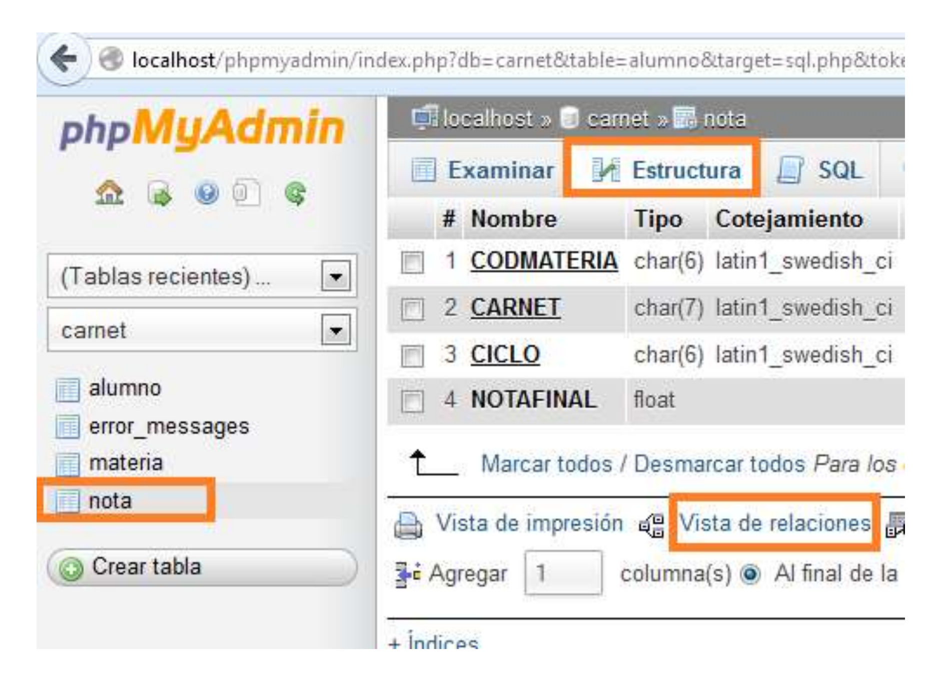
En Nota

INSERT INTO NOTA VALUES('MAT115', 'NN00001','1',2);

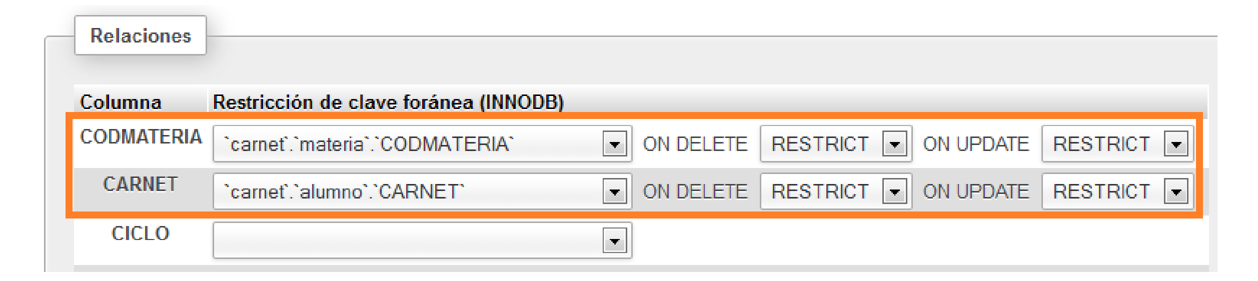
INSERT INTO NOTA VALUES('MAT115', 'NN00001','1',2);



1) Verificamos en la tabla nota. Que existe la integridad Referencial(Clic en tabla NOTAS, clic en Estructura y clic en vista relaciones)



Veremos que efectivamente están activadas las restricciones de Foreign Key para modalidad RESTRICT en OPERACIÓN UPDATE Y DELETE.



Aunque no aparezca, Las restricciones anteriores son válidas para INSERT.

2) Probar integridad referencial de nota con las tablas Materia y Alumno

Probamos la integridad referencial (con un carnet que no tiene referencia en la tabla alumno)

INSERT INTO NOTA VALUES('MAT115', 'NN00002','1',2);

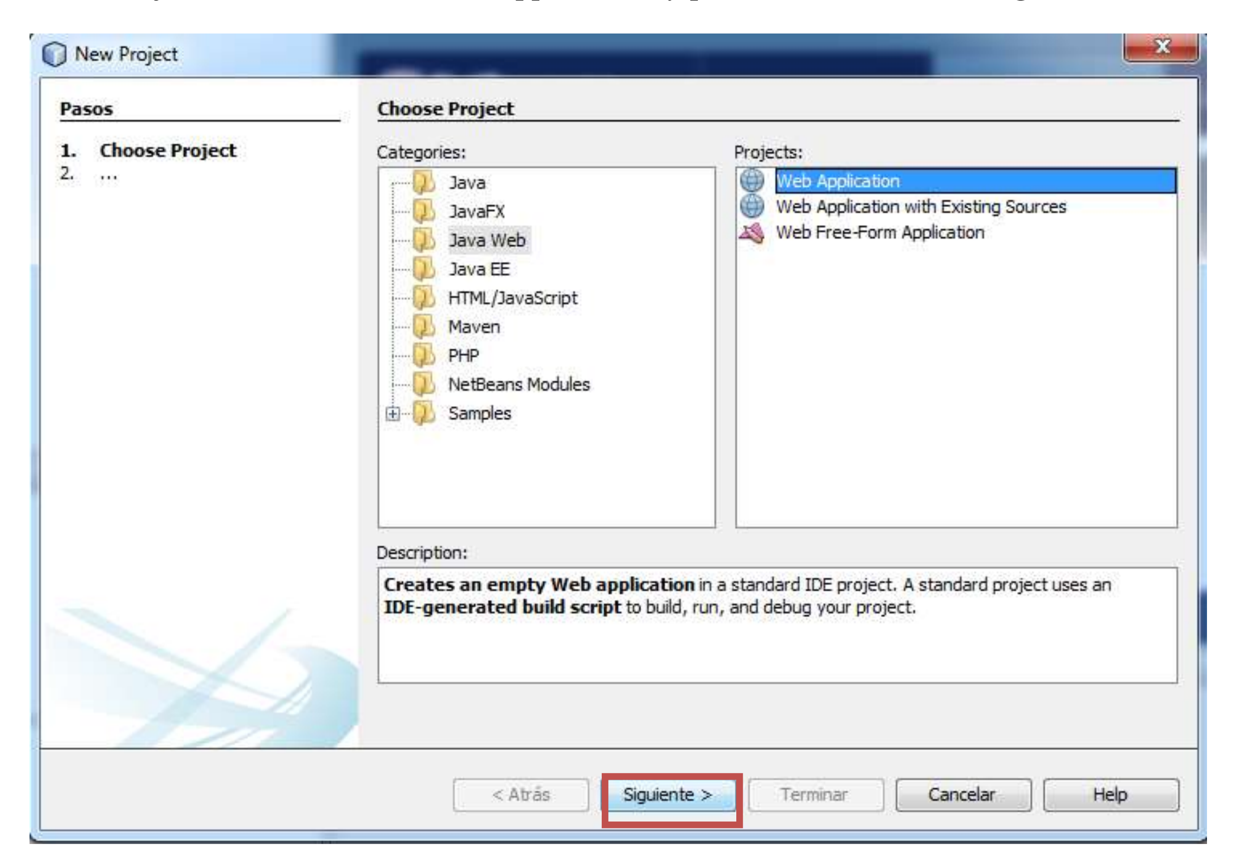
Probamos la integridad referencial (con un codmateria que no tiene referencia en la tabla materia)

INSERT INTO NOTA VALUES('MAT215', 'NN00001','1',2);

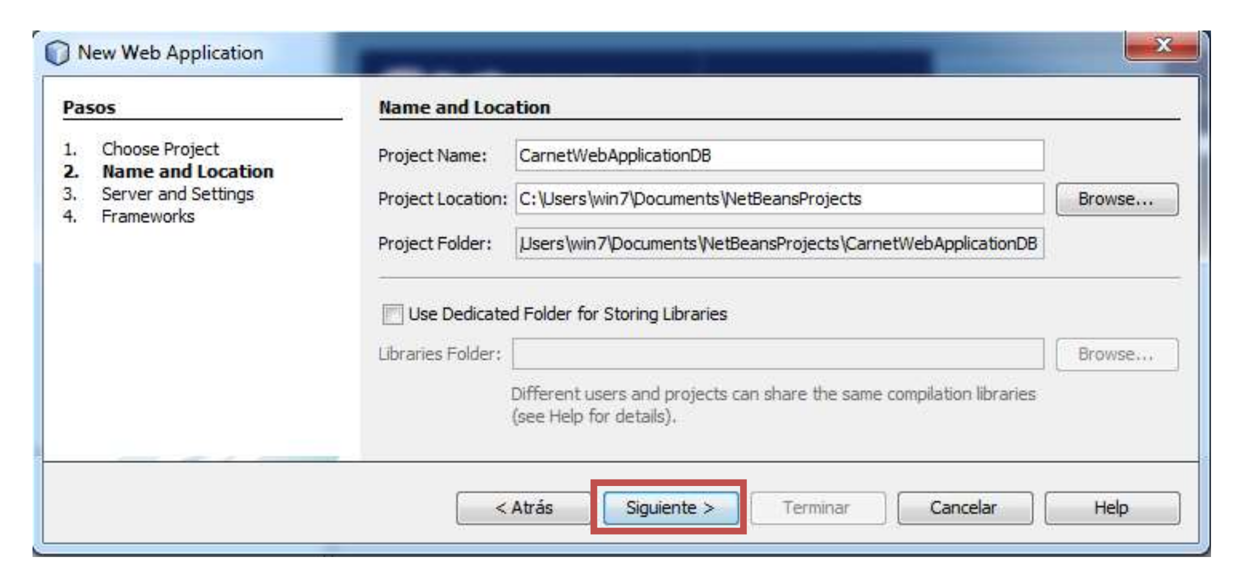
Creación de Servicios web(con asistentes)

1. Crear el Proyecto

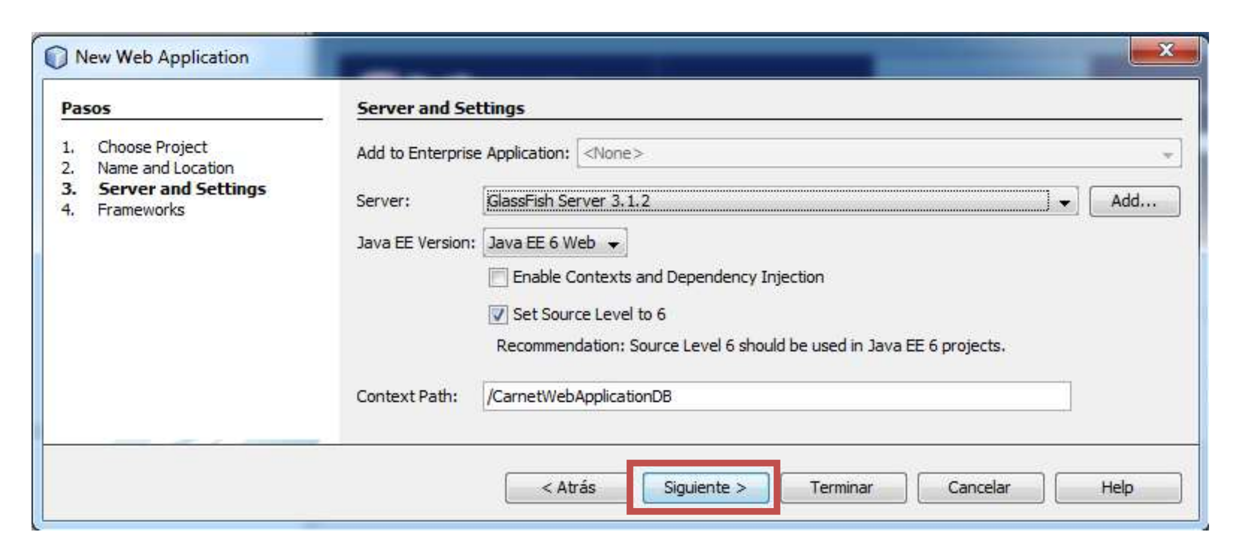
Como primer paso ejecutamos NetBeans IDE y dentro del menú "File" seleccionamos "New Project", "Java Web", "Web Application" y presionamos el botón "Siguiente":



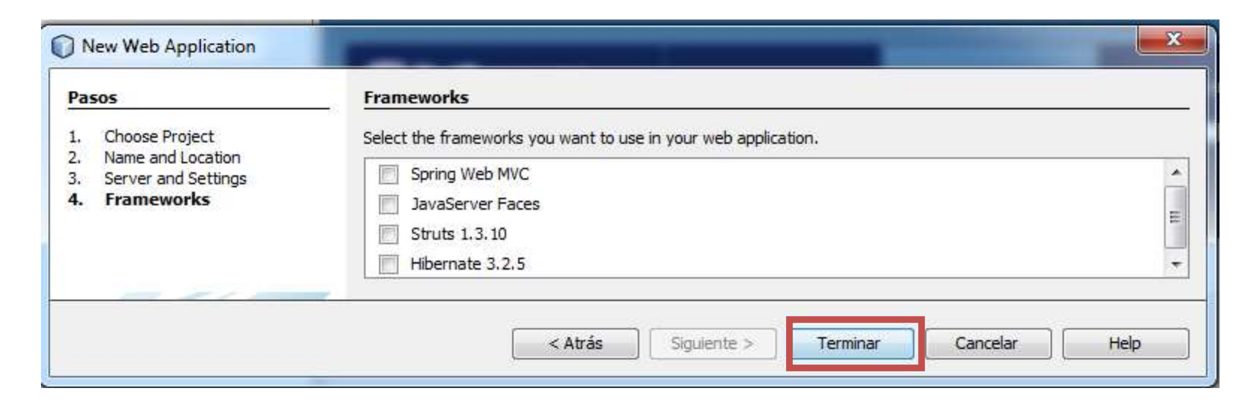
En la siguiente ventana establecemos el nombre(CarnetWebApplicationDB) y la ubicación del proyecto, luego presionamos el botón “Siguiente”:



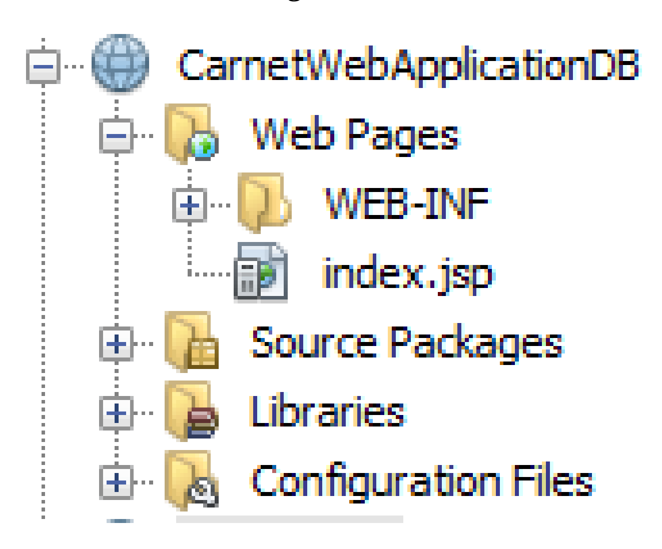
Continuamos sin alterar las configuraciones del Servidor y versión de javaEE. Presione Siguiente.



Por último podemos seleccionar un Framework, para nuestro caso dentro del desarrollo de esta guía no aplica, por lo tanto presionamos el botón "Terminar"



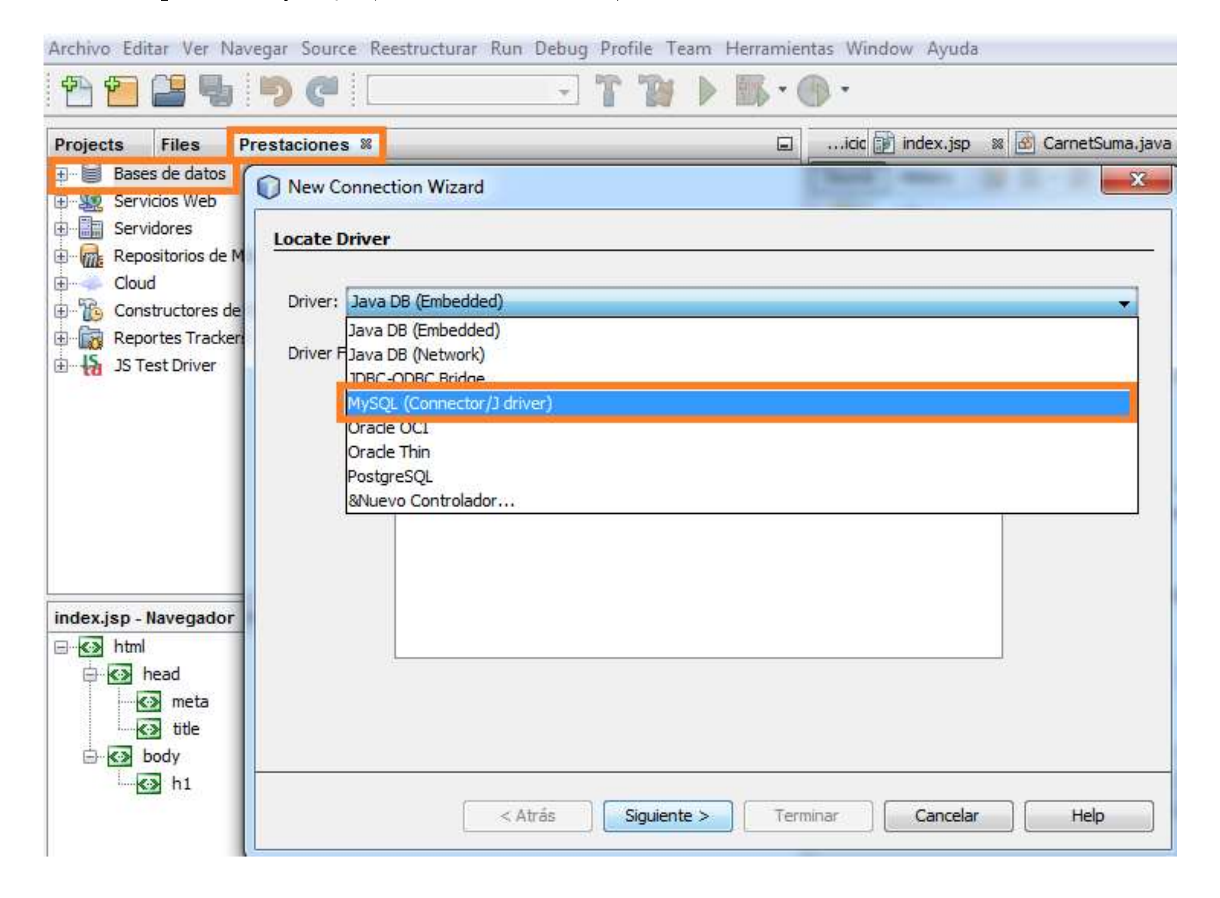
Una vez creado tendremos la siguiente estructura dentro de la vista de Proyectos:



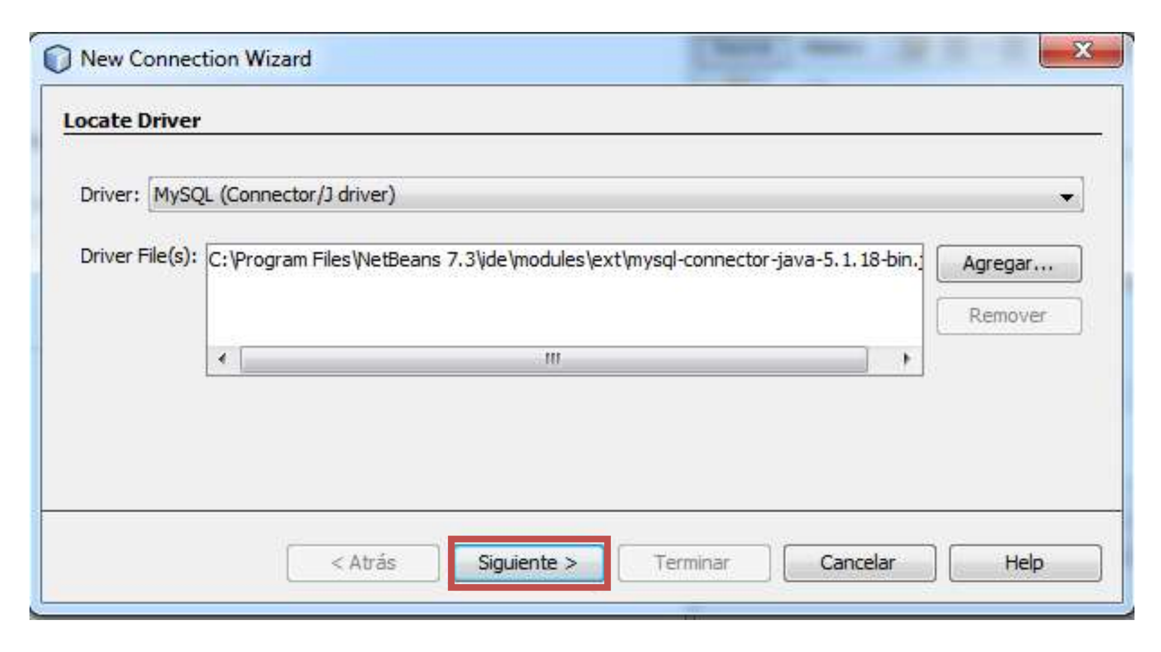
2. Conectando a la Base de Datos

Ahora crearemos la conexión a nuestra base de datos CARNET en MySQL, dentro del panel del proyecto iremos a la pestaña "Prestaciones" y al hacer click derecho sobre "Bases de Datos" seleccionamos "New Connection"

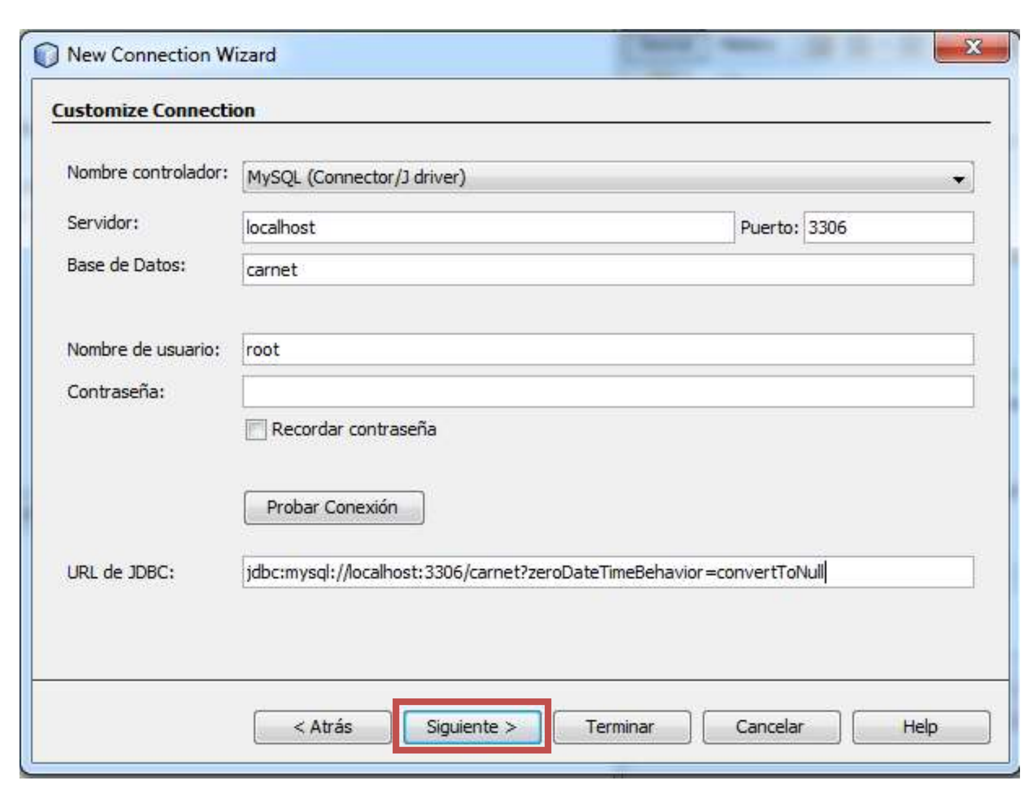
Se desplegará un asistente para la nueva conexión, en la opción "Driver" seleccionamos en la lista la opción "MySQL (Connector/J driver)"



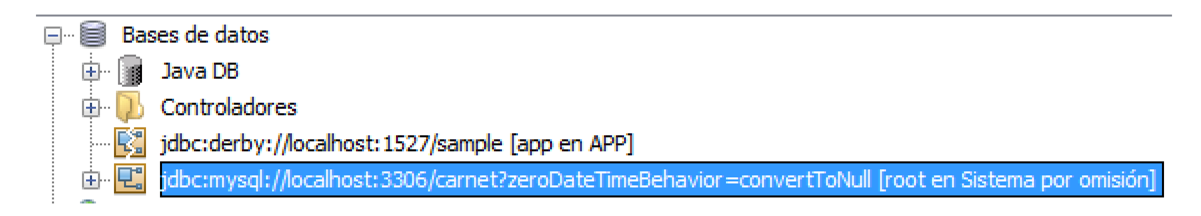
Una vez seleccionado MySQL presionamos sobre el botón "Siguiente":



En la siguiente ventana establecemos los valores para la conexión a la Base de Datos, el nombre, usuario y password. Para comprobar que los valores son correctos presionamos el botón "Test Connection", una vez correcta la conexión presionamos sobre el botón "Finish"

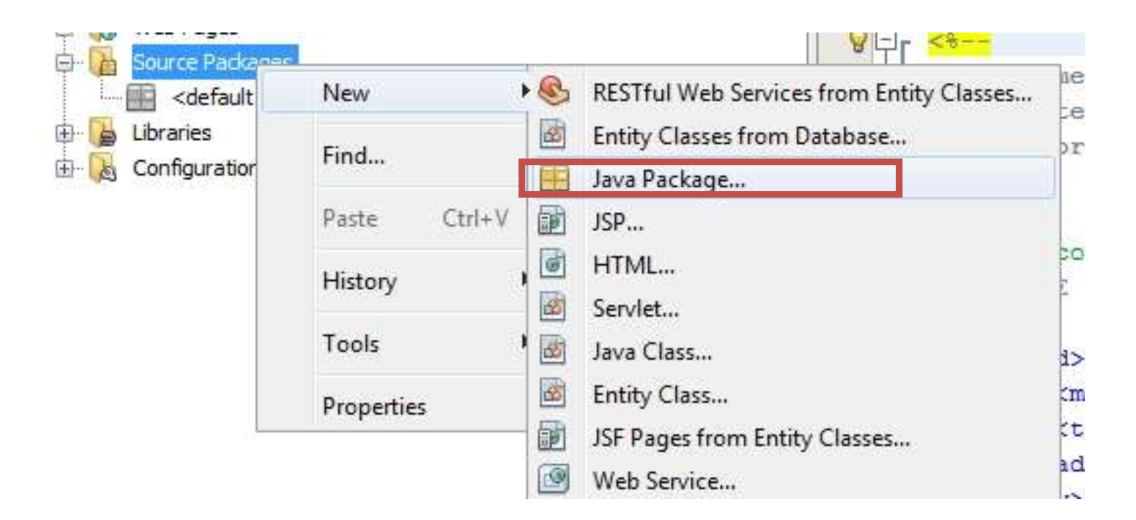


Una vez creada la conexión se puede visualizar dentro de la pestaña "Prestaciones"

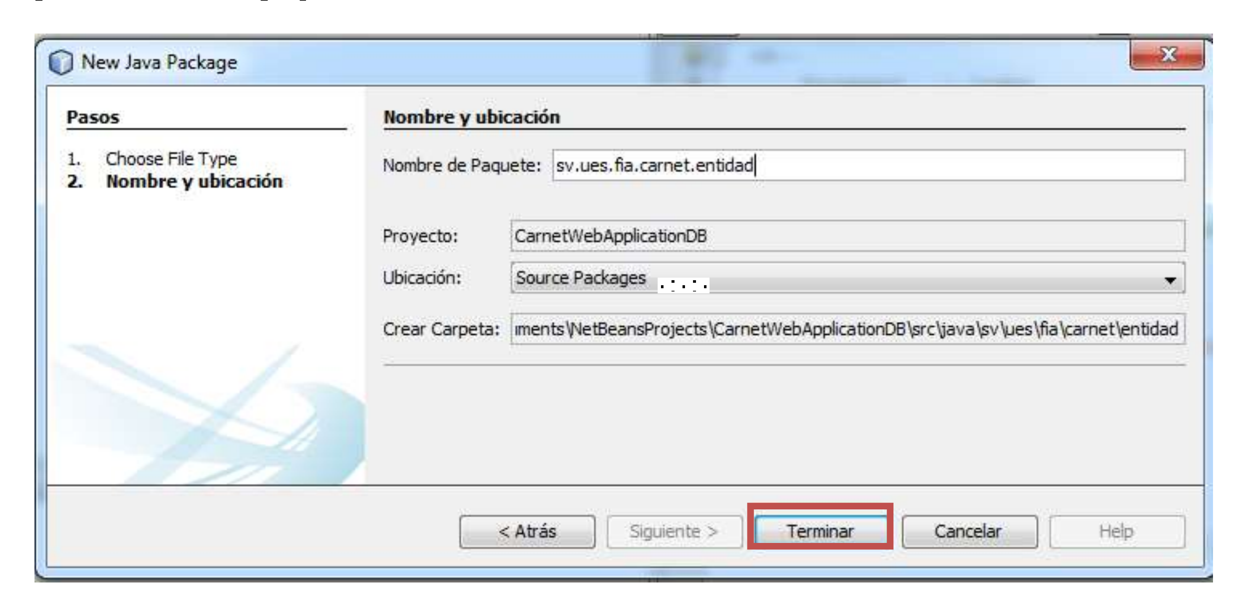


3. Creación de las Clases Entidad

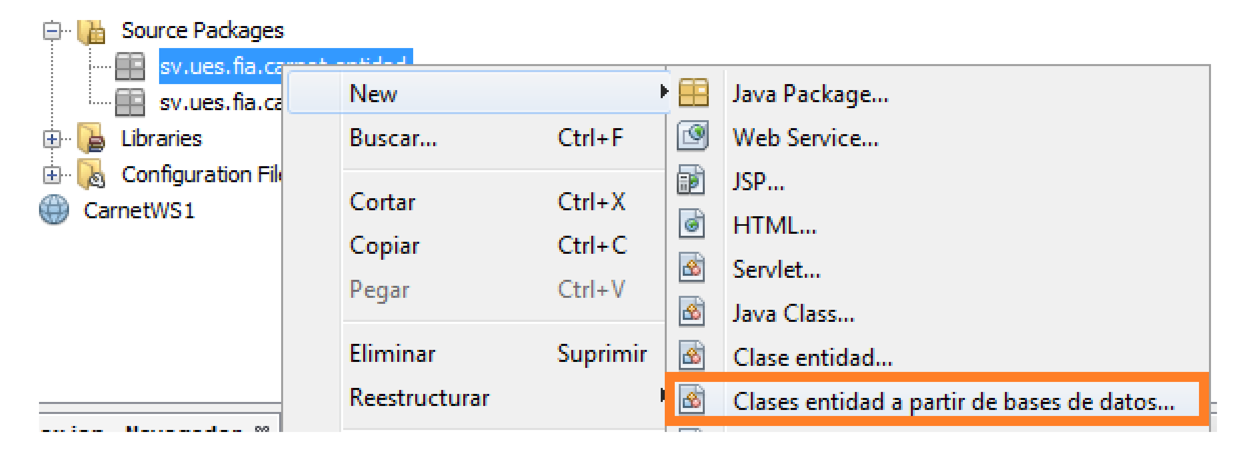
Ahora dentro de la pestaña "Projects" crearemos dos paquetes, el primero llamado "sv.ues.fia.carnet.entidad" contendrá las clases entidad generadas de nuestra base de datos, el segundo "sv.ues.fia.carnet.servicio" tendrá las clases que contienen los servicios web. Para ello hacemos click derecho sobre "Source Packages" → "New" → "Java Package"



Establecemos el nombre y presionamos el botón "Terminar", repetimos este procedimiento para crear el otro paquete.

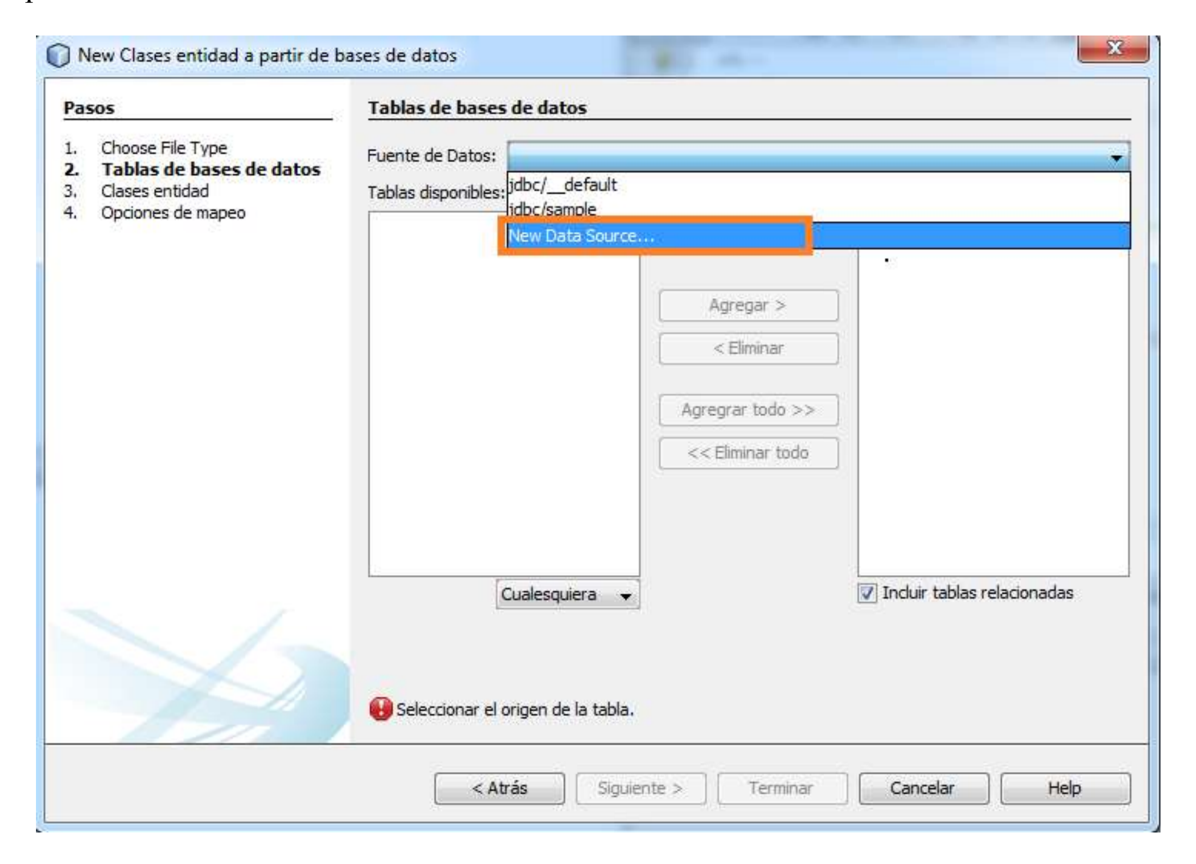


Ahora hacemos click derecho sobre el paquete "sv.ues.fia.carnet.entidad" y buscamos la opción "New" → "Clases entidad a partir de bases de datos"

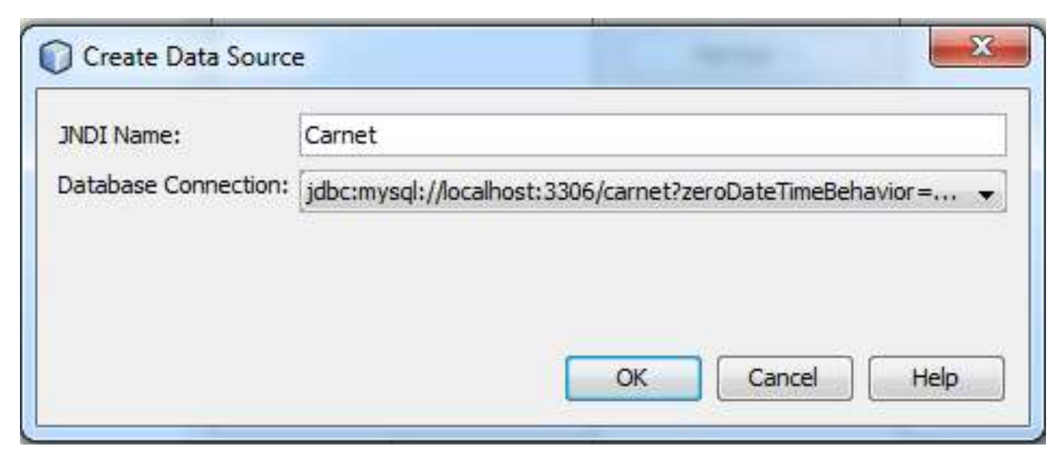




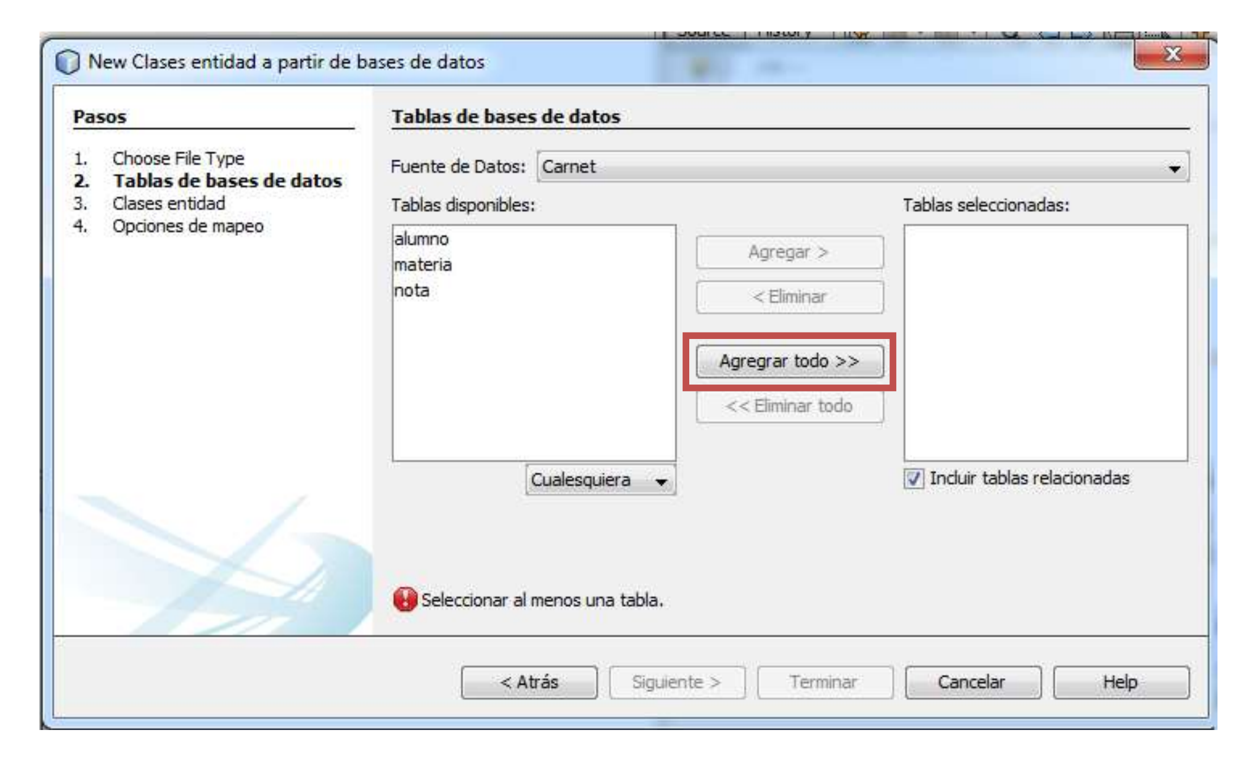
Después de seleccionar “Clases entidad a partir de bases de datos” se muestra la siguiente ventana donde debemos indicar la conexión que corresponde a nuestra Base de Datos, en la opción “Fuente de Datos” seleccionamos “New Data Source”



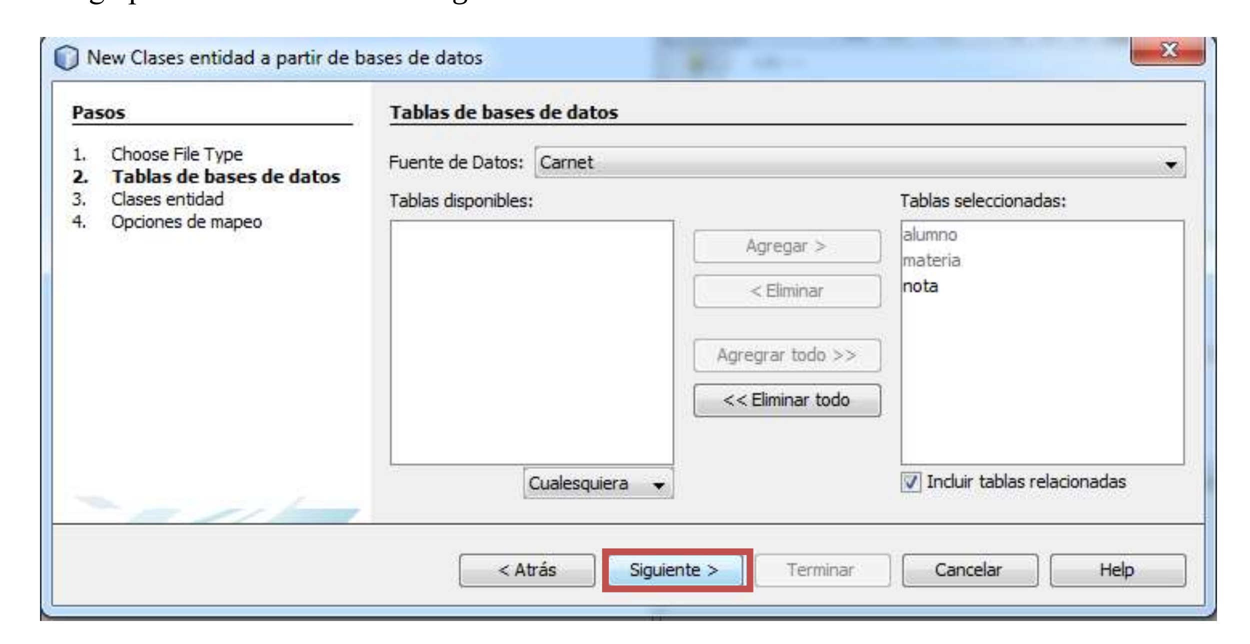
En el siguiente cuadro colocamos Carnet en la opción “JNDI Name”, y en “Database Connection” deberá aparecernos la que fue creada con anterioridad, la seleccionamos y presionamos el botón “OK”



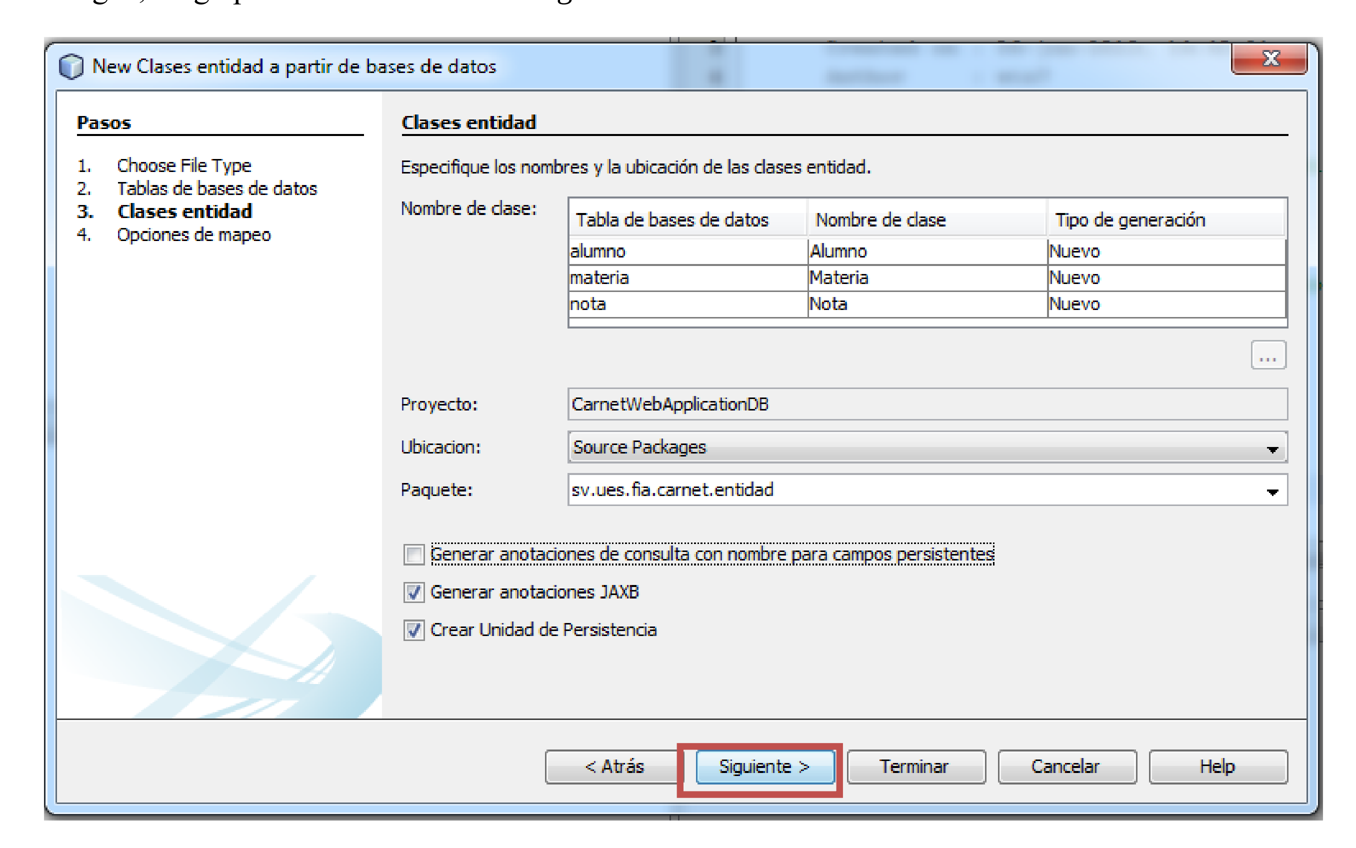
Una vez realizado podemos observar que aparecerán las tablas de nuestra base, presionamos el botón “Agregar todo” para crear las clases de todas las tablas



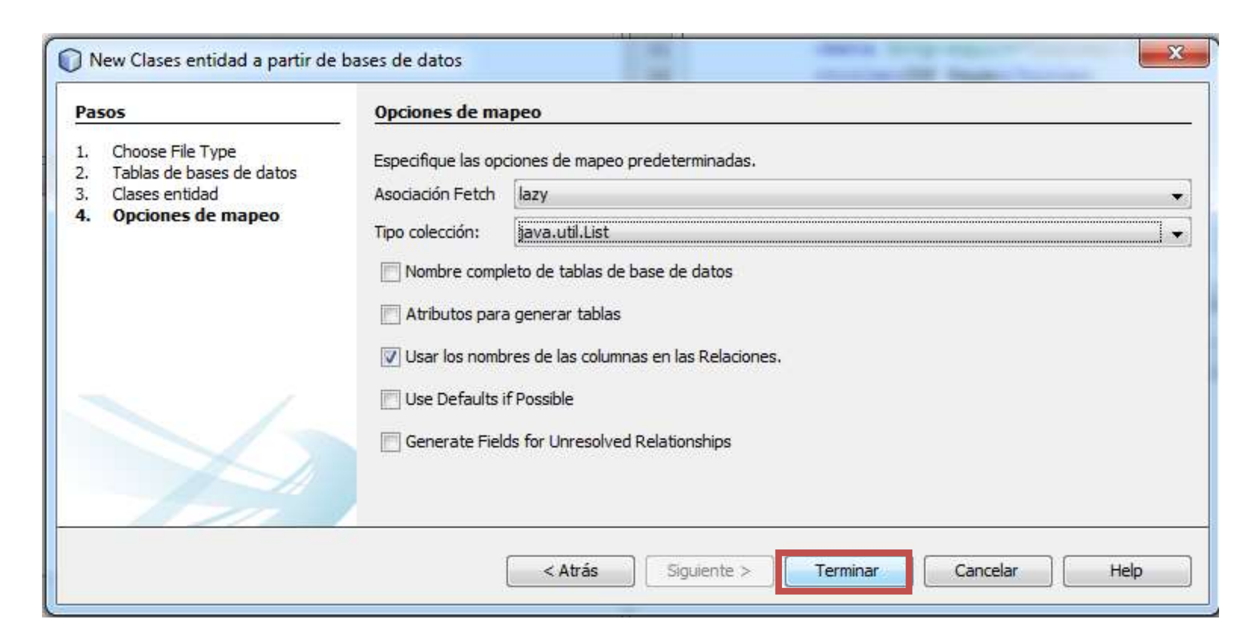
Luego presionamos el botón “Siguiente”



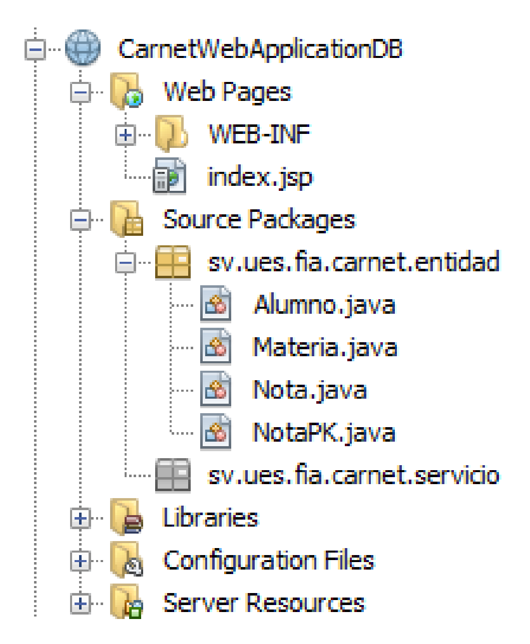
En la siguiente ventana verificamos que el paquete que contendrá nuestras clases sea "sv.ues.fia.carnet.entidad", y estén seleccionadas solamente las opciones que muestra la imagen, luego presionamos el botón "Siguiente".



Por último especificamos las opciones del mapeo de entidades, en la opción "Association Fetch" seleccionamos "lazy" y en "Collection Type" seleccionamos "java.util.List" y las demás opciones tal como muestra la imagen. Para finalizar presionamos el botón "Terminar"

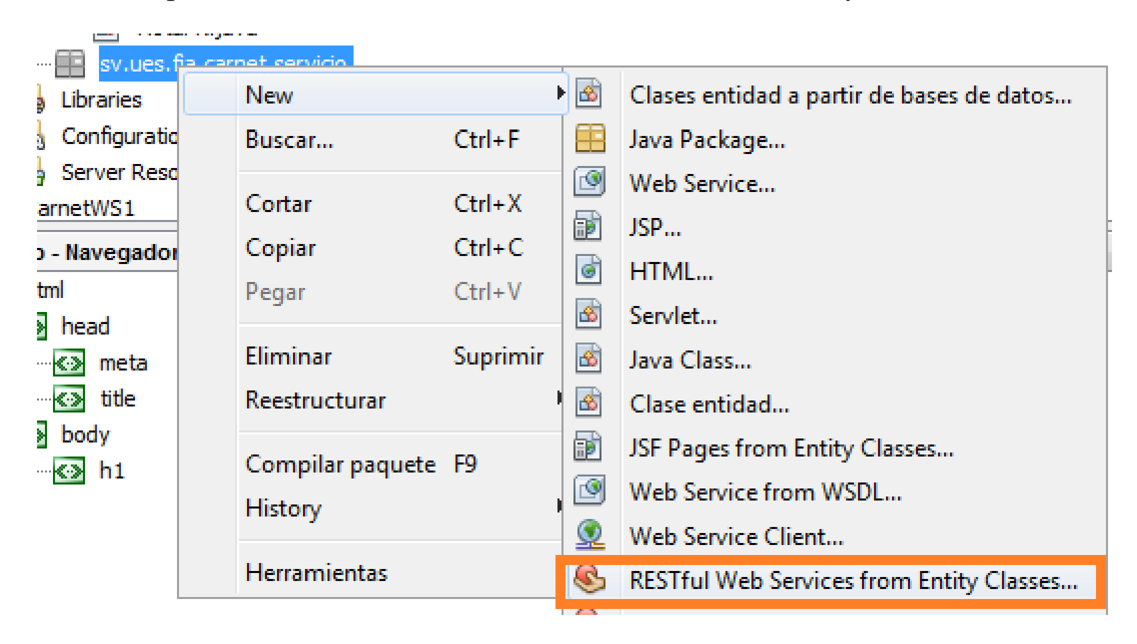


Una vez completado podemos observar las Clases creadas dentro de nuestro paquete, como se puede observar cada una representa la abstracción de la correspondiente tabla con sus campos, la clase NotaPK es generada ya que la llave primaria de nuestra tabla Nota está compuesta por 3 campos (Carnet, CodMateria y Ciclo), y es necesario englobar dicha llave primaria en un solo objeto que contenga dichos campos.

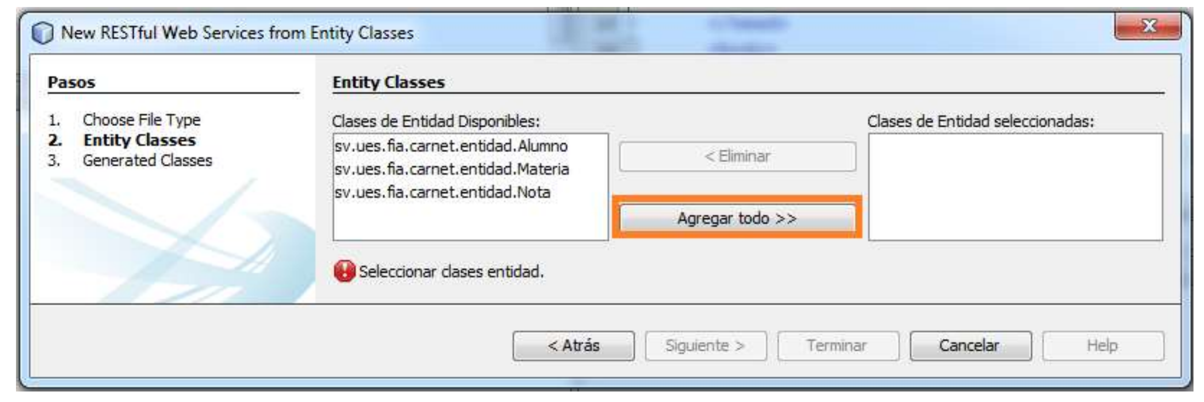


4. Creación de los servicios web

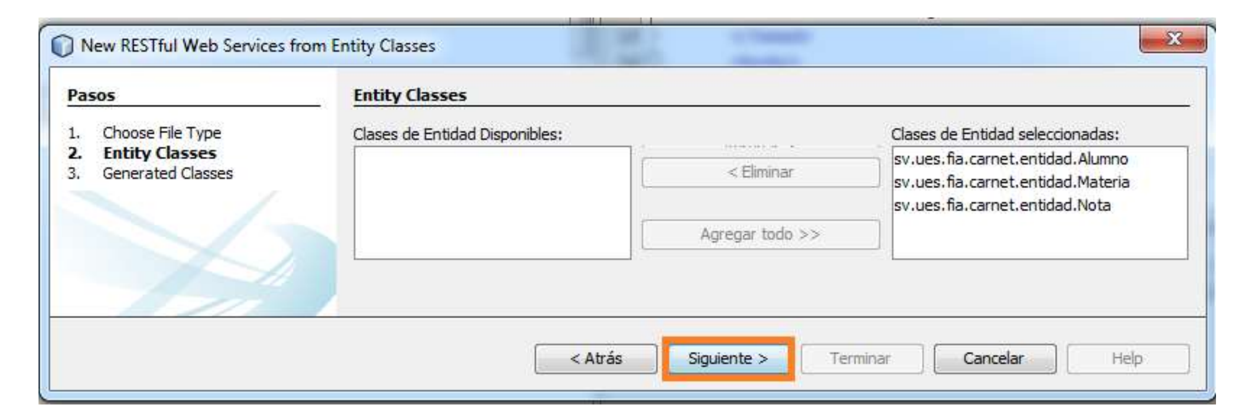
Ahora crearemos las clases controladoras que implementan los servicios web, para ello seleccionamos nuestro paquete "sv.ues.fia.carnet.servicio" y hacemos click derecho, luego buscamos la opción "New" → "RESTful Web Services from Entity Classes"



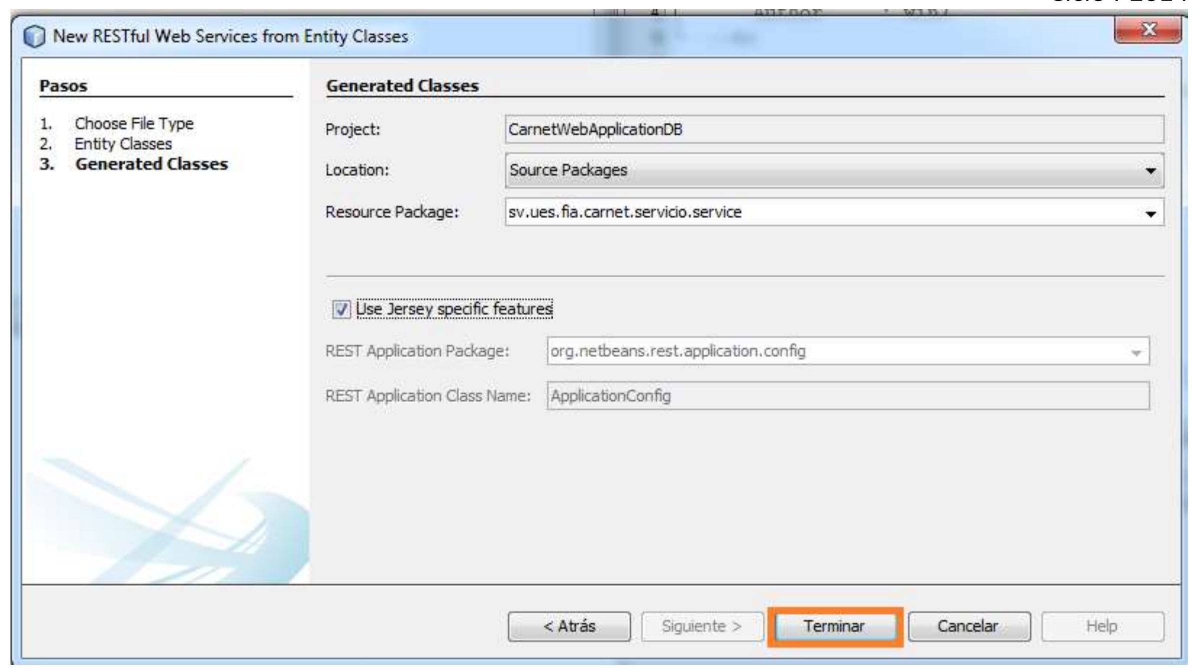
Presionamos "Next" y nos mostrará la siguiente ventana donde se muestras las clases entidad disponibles, presionamos el botón "Agregar todo" para generar los servicios de todas las clases



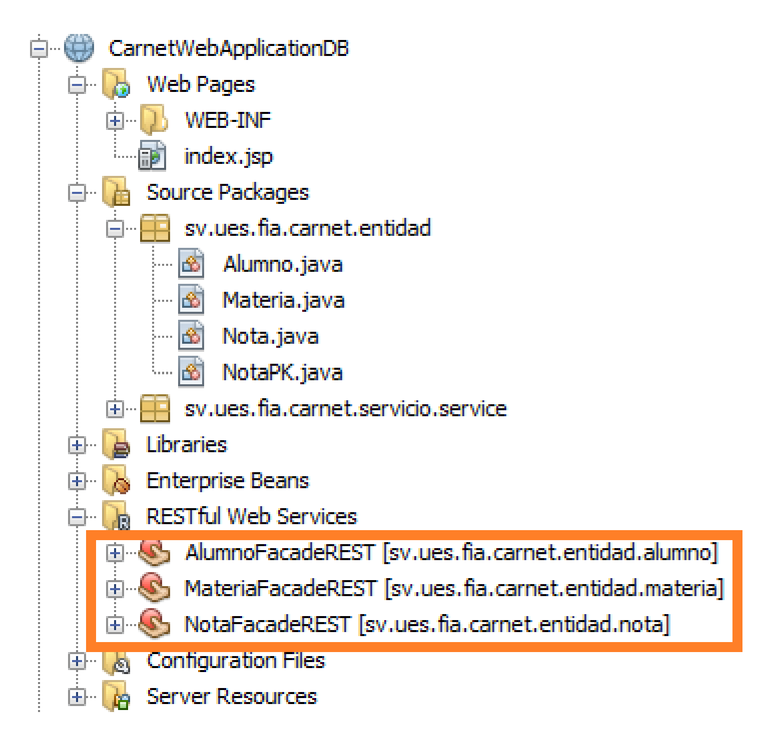
Luego presionamos el botón “Siguiente”



En la siguiente ventana verificamos que nuestros servicios sean creados en el paquete especificado “sv.ues.fia.carnet.servicio”, y seleccionamos la opción “Use Jersey specific features”, para finalizar presionamos el botón “Terminar”.

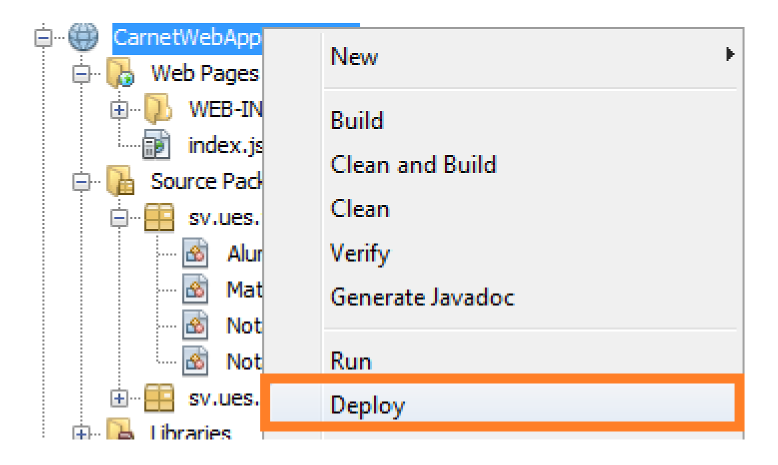

New RESTful Web Services from Entity Classes
Pasos
1. Choose File Type
2. Entity Classes
3. Generated Classes
Generated Classes
Project:
CarnetWebApplicationDB
Location:
Source Packages
Resource Package:
sv.ues.fia.carnet.servicio.service
Use Jersey specific features
REST Application Package:
org.netbeans.rest.application.config
REST Application Class Name:
ApplicationConfig
< Atrás
Siguiente >
Terminar
Cancelar
Help

Una vez creados los servicios web podemos observar las clases controladoras dentro de la vista del proyecto.



5. Probando los servicios web

Hasta este punto ya tenemos todo listo para probar los servicios web, para ello primero es necesario desplegar la aplicación dentro del servidor, para ello seleccionamos el proyecto haciendo click derecho sobre él y buscando la opción “Deploy”



Esperamos mientras carga el servidor GlassFish y se despliegan los servicios, podemos consultar la consola del servidor para verificar el correcto despliegue:

```
Salida
CarnetWebApplicationDB (run-deploy) | Proceso de bases de datos Java DB | GlassFish Server 3.1.2
INFO: Initiating Jersey application, version 'Jersey: 1.11.1 03/31/2012 06:49 PM'
INFO: Binding the EJB class sv.ues.fia.carnet.servicio.service.AlumnoFacadeREST to EJBManagedComponentProvider
INFO: Binding the EJB class sv.ues.fia.carnet.servicio.service.MateriaFacadeREST to EJBManagedComponentProvider
INFO: Binding the EJB class sv.ues.fia.carnet.servicio.service.NotaFacadeREST to EJBManagedComponentProvider
INFO: WEB0671: Loading application [CarnetWebApplicationDB] at [/CarnetWebApplicationDB]
INFO: CarnetWebApplicationDB se ha desplegado correctamente en 5.043 milisegundos.
```

Instalar Plugin de Firefox

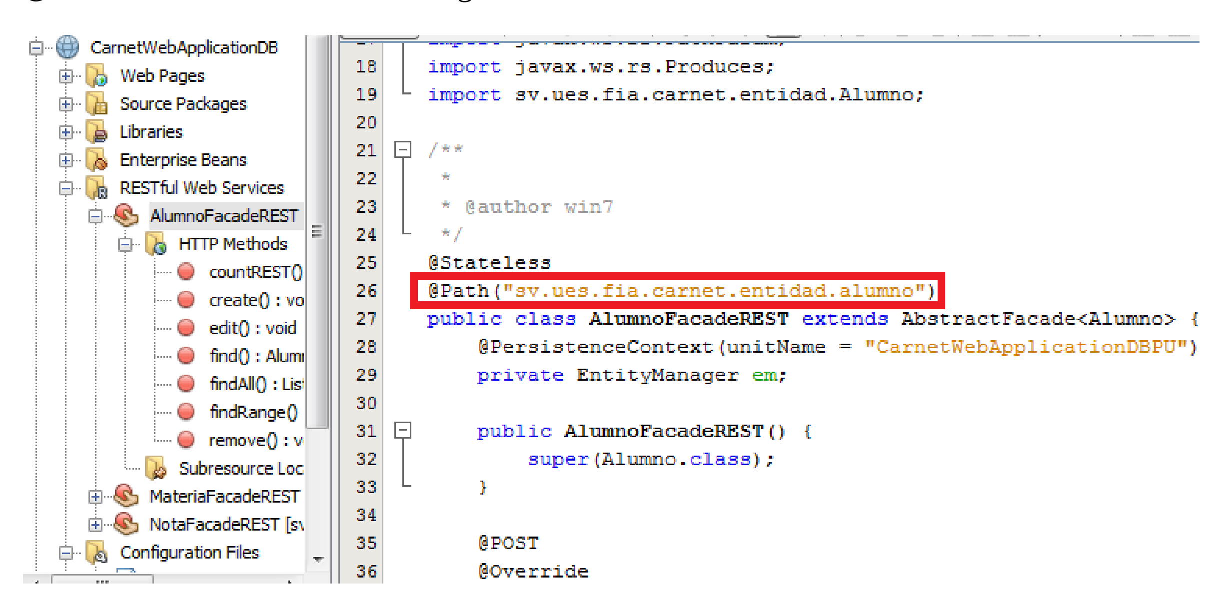
Para verificar el funcionamiento de los servicios web podemos acceder desde un navegador a las URL que se presentan para cada caso, con esto obtendremos los datos en formato XML, sin embargo existen herramientas especiales que nos permiten una mejor depuración para este tipo de servicios, en esta guiá se utiliza la extensión para el navegador Firefox llamada "REST Client" (https://addons.mozilla.org/es/firefox/addon/restclient/) esta herramienta también nos permite establecer otras opciones de petición de datos a los servicios web, como por ejemplo en lugar de obtener los datos en formato XML los podemos obtener en formato JSON.

La URL de los servicios se especifica mediante el archivo de configuración **web.xml** tal como muestra las siguientes imágenes:

```
CarnetWebApplicationDB
  Web Pages
  Source Packages
  Libraries
  Enterprise Beans
  RESTful Web Services
  Configuration Files
    MANIFEST.MF
    persistence.xml
    web.xml
  Server Resources
```

```
<servlet-mapping>
    <servlet-name>ServletAdaptor</servlet-name>
    <url-pattern>/webresources/*</url-pattern>
</servlet-mapping>
<session-config>
    <session-timeout>
        30
    </session-timeout>
</session-config>
```

Y para cada controlador de servicios se especifica su dirección mediante la anotación @Path como se muestra en la imagen:



A continuación se presenta un conjunto de operaciones como ejemplo:

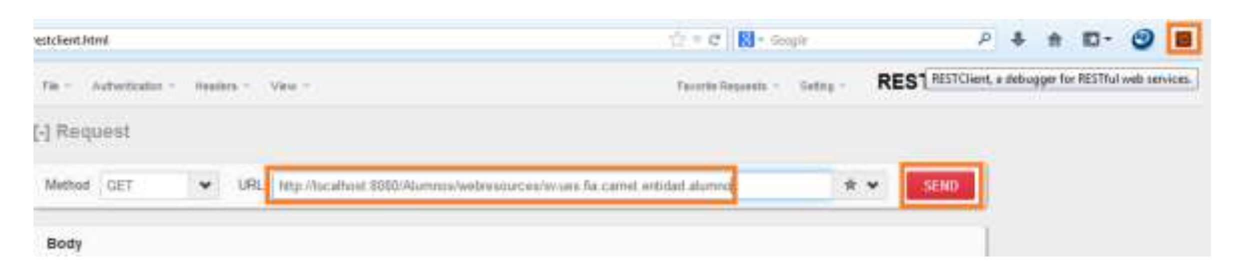
Obtener los datos de todos los alumnos en XML

Cargamos el Firefox y entramos al plugin que acabamos de instalar

Entramos los parámetros:

Método: GET

URL: http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno



Presionamos el botón "SEND"

Veremos primeramente la cabecera de la respuesta

[-] Response

Response Headers | Response Body (Raw) | Response Body (Highlight) | Response Body (Preview)

```
Status Code     : 200 OK
Content-Length  : 206
Content-Type    : application/xml
Date            : Thu, 06 Jun 2013 22:46:51 GMT
Server          : GlassFish Server Open Source Edition 3.1.2.2
X-Powered-By    : Servlet/3.0 JSP/2.2 (GlassFish Server Open Source Edition 3.1.2.2 Java/Oracle Corporation/1.7)
```

Primero podemos observar la cabecera de la respuesta (Response) HTTP, para los datos obtenidos accedemos a la pestaña "Response Body (Highlight)".

[-] Response

Response Headers | Response Body (Raw) | Response Body (Highlight) | Response Body (Preview)


```xml
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<alumnoes>
  <alumno>
    <apellido>Nuevo</apellido>
    <carnet>NN00001</carnet>
    <matganadas>0</matganadas>
    <nombre>Nuevo</nombre>
    <sexo>77</sexo>
  </alumno>
</alumnoes>
```

Obtener los datos de un alumno por medio de su carnet en JSON:

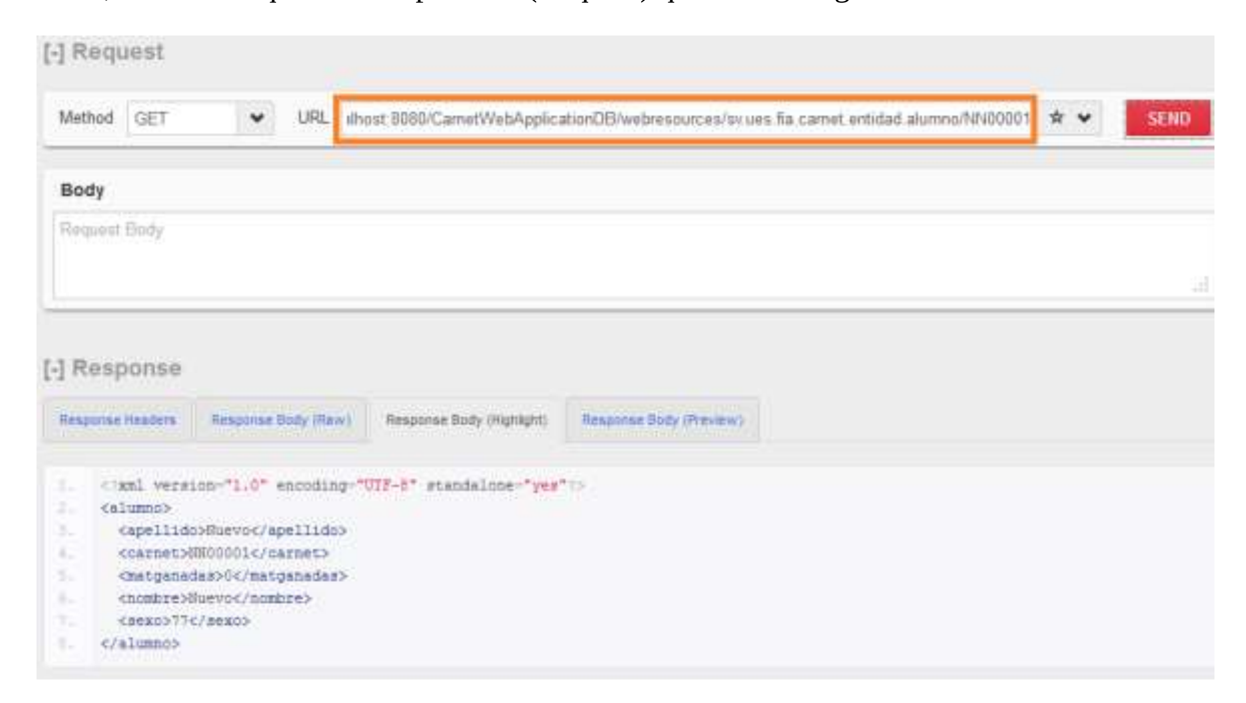
Para obtener los datos en formato JSON es necesario especificarlo dentro de la cabecera de la petición HTTP:

Cambie los parámetros:

Método: GET

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/NN00001

En la parte final de la URL especificamos el carnet del alumno que deseamos obtener sus datos, de manera que nuestra petición (Request) queda de la siguiente manera:



Al ejecutar la petición obtenemos el Response con los siguientes datos:

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<alumno>
  <apellido>Nuevo</apellido>
  <carnet>NN00001</carnet>
  <matganadas>0</matganadas>
  <nombre>Nuevo</nombre>
  <sexo>77</sexo>
</alumno>
```

De la misma manera ejecutamos las siguientes pruebas con las URL's indicadas:

Obtener los datos de todas las materias

Método: GET

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.materia

Obtener los datos de una materia por medio de su código;

Método: GET

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.materia/MAT115

En la parte final de la URL especificamos el código de la materia que deseamos obtener sus datos.

Obtener la nota de un alumno para una materia y ciclo dado:

Método: GET

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.nota/id;carnet=NN00001;codmateria=MAT115;ciclo=1

En la parte final de la URL especificamos los tres valores necesarios para obtener la nota de un alumno, su carnet, código de materia y el ciclo en que la curso.

Guardar los datos de un nuevo alumno

Como primer paso modificaremos el método que recibe la petición POST (create) de la clase AlumnoFacadeREST para que nos devuelva una respuesta favorable si la inserción se realizó de manera correcta.

```
@POST
@Consumes({"application/xml", "application/json"})
public Response createAlumno(Alumno entity) {
    super.create(entity);
    return Response.ok().build();

}
```

Es necesario renombrar el método create ya que la clase hereda de AbstractFacade y está ya contiene dicho método create pero sin retornar valores.

Verificar que el import necesario para la clase Response sea el siguiente:

```
import javax.ws.rs.core.Response;
```

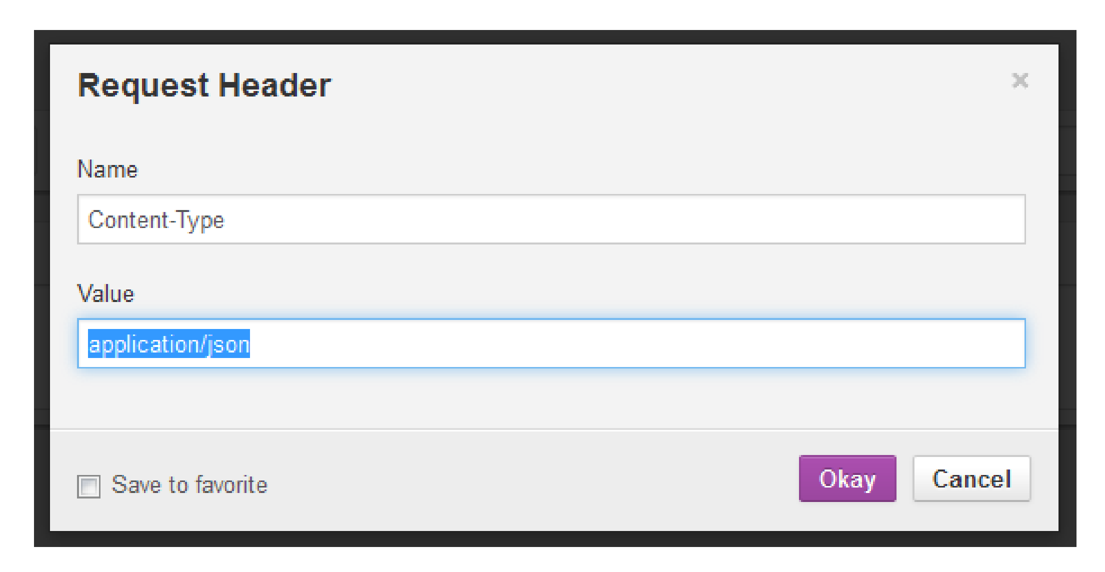
Guardamos el proyecto y desplegamos nuevamente la aplicación.

Para almacenar datos desde un servicio web es necesario realizar una petición POST, la cual debe contener los datos del nuevo alumno ya sea en formato XML o JSON, a continuación se presenta ejemplos utilizando el formato JSON.

Método: POST

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/

En la petición se creara un nuevo encabezado HTTP con la opción **Content-type: application/json**

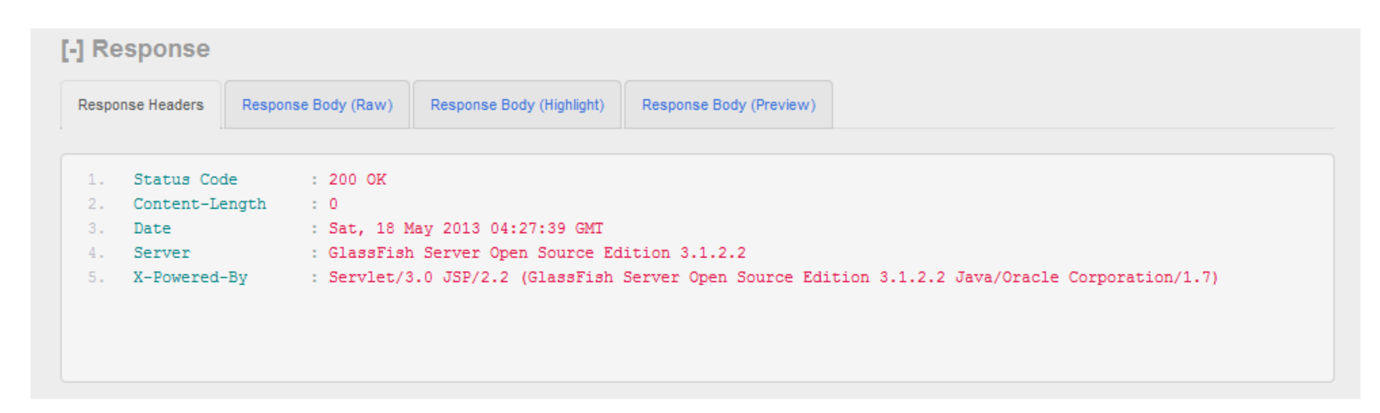


Colocamos dentro del cuerpo de la petición los datos del alumno en formato JSON, al final nuestra petición quedará de la siguiente manera:

```
{
"carnet":"NN00002",
"nombre":"carlos",
"apellido":"perez",
"sexo":"M",
"matganadas":0
}
```



Presionamos sobre el botón SEND y obtenemos la respuesta del servidor, en este caso el alumno se guardó correctamente



Si el servidor encuentra algún error al momento de guardar el alumno, como por ejemplo una duplicación de carnet, el resultado sería como el siguiente:



De la misma forma se puede implementar para Materia y Nota sus correspondientes métodos POST.

Guardar los datos de una nueva materia

Método: POST

URL: http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.materia

{"codmateria":"PDM115","nommateria":"Programacion para DM","unidadesval": “8”}



Con esta petición se crea una nueva materia en la base de datos del servidor, se pueden modificar el resto de métodos para que devuelvan un código de estado 200 (que significa OK) si la inserción se realiza de manera correcta, tal como se hizo con la petición POST para alumnos. Si no se modifica, la respuesta por defecto del servidor es la siguiente:



Guardar las notas de un alumno

Método: POST

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.nota/

{

"notaPK":

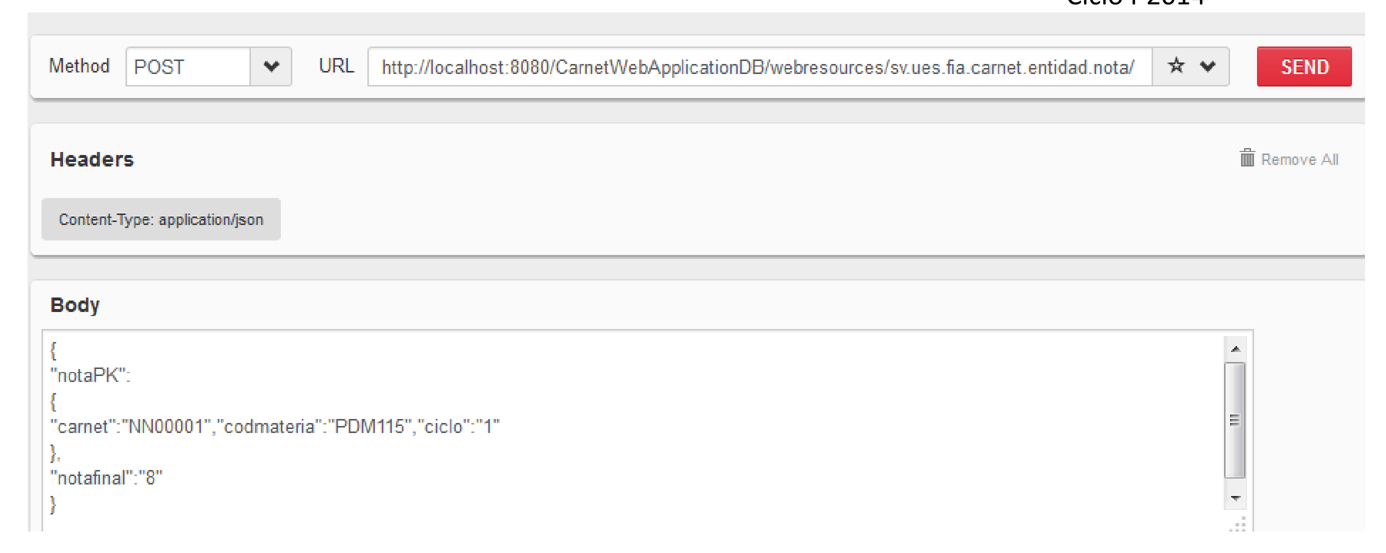
{

"carnet":"NN00001","codmateria":"PDM115","ciclo":"1"

},

"notafinal":"8"

}


Method
POST
URL
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.nota/
SEND
Headers
Remove All
Content-Type: application/json
Body
{
"notaPK":
{
"carnet":"NN00001","codmateria":"PDM115","ciclo":"1"
},
"notafinal":"8"
}

Modificar los datos de un alumno

La modificación de registros existentes se realizan mediante peticiones PUT al servidor web, para ello dentro del contenido de la petición es necesario colocar los datos del alumno a modificar, para este ejemplo se cambió el email del alumno con carnet NN00001.

Método: PUT

Realice los cambios de Response a todos los métodos PUT(tablas alumno, materia y nota)

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/

{

"carnet":"NN00001",

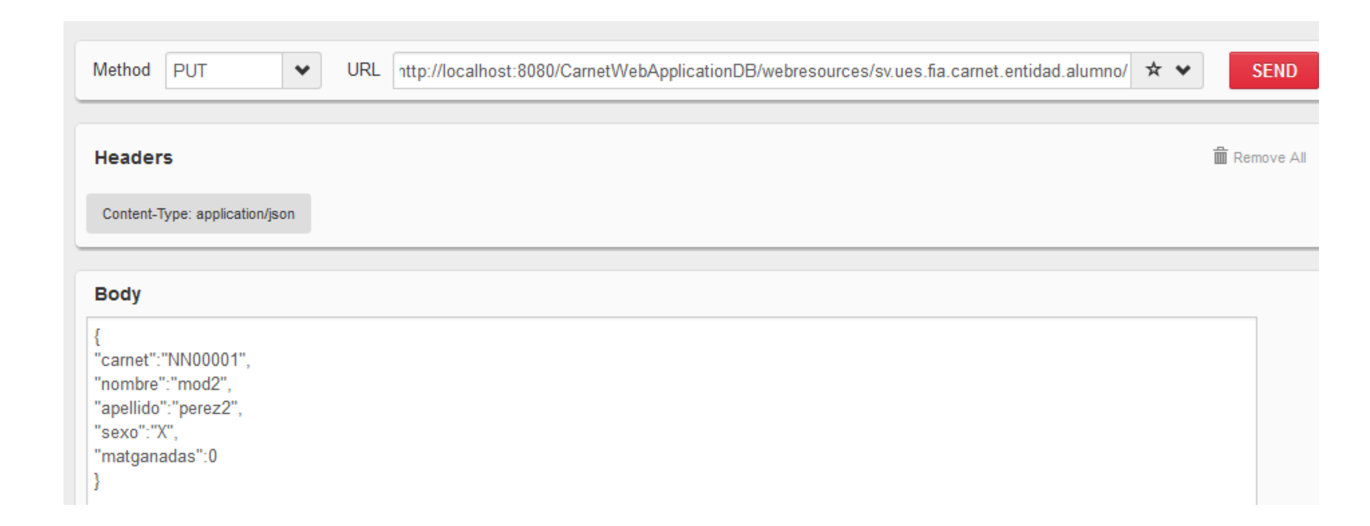
"nombre":"mod3",

"apellido":"perez3",

"sexo":"X",

"matganadas":0

}



Modificar los datos de una Nota

El nombre de la materia “Matematicas 1” se cambió a “Mate 1”

Método: PUT

URL: http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.nota/

{

"notaPK":

{

"carnet":"NN00001","codmateria":"PDM115","ciclo":"1"

},

"notafinal":"8"

}

Method PUT URL http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.nota/ SEND

Headers Remove All

Content-Type: application/json

Body

```
{
"notaPK":
{
"carnet":"NN00001","codmateria":"PDM115","ciclo":"1"
},
"notafinal":"6"
}
```

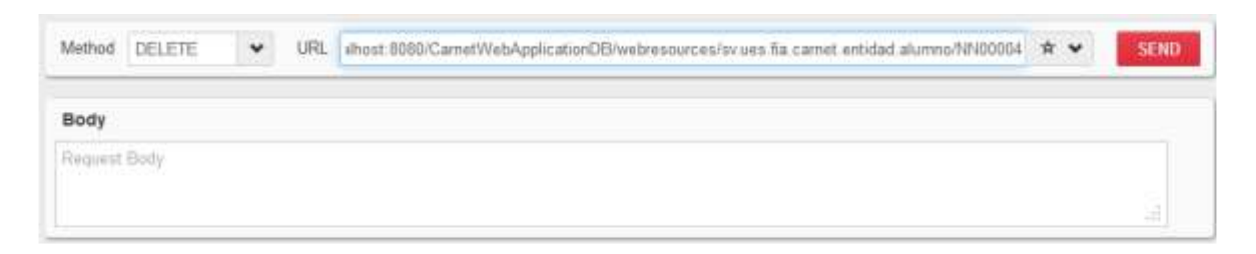
Eliminar los datos de un alumno

Método: DELETE

Realice los cambios de Response a todos los métodos DELETE(tablas alumno, materia y nota solo que sin cambiar los nombres de método ya que no hay @override)

URL:
http://localhost:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/NN00004

En la parte final de la URL especificamos el carnet del alumno que deseamos eliminar sus datos, para este ejemplo eliminaremos el alumno recién creado con carnet NN00004 ya que no contiene ningún registro de notas y no viola la integridad referencial.



De igual manera se puede proceder a la eliminación de registros de materias y notas pero teniendo en consideración las implicaciones de dicha eliminación.

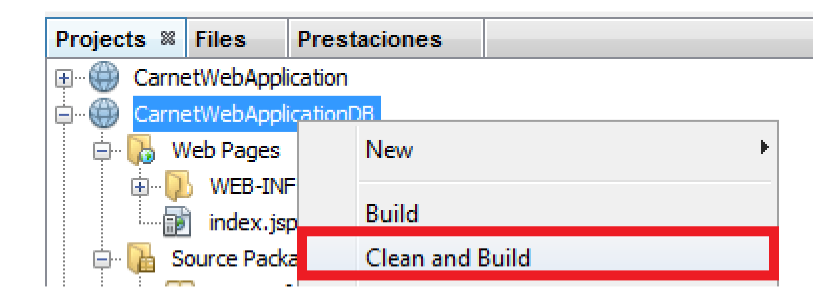
Cuando finalice la práctica solo envié el

***El archivo comprimido del proyecto netbeans de practica 9b_Proyecto.rar**

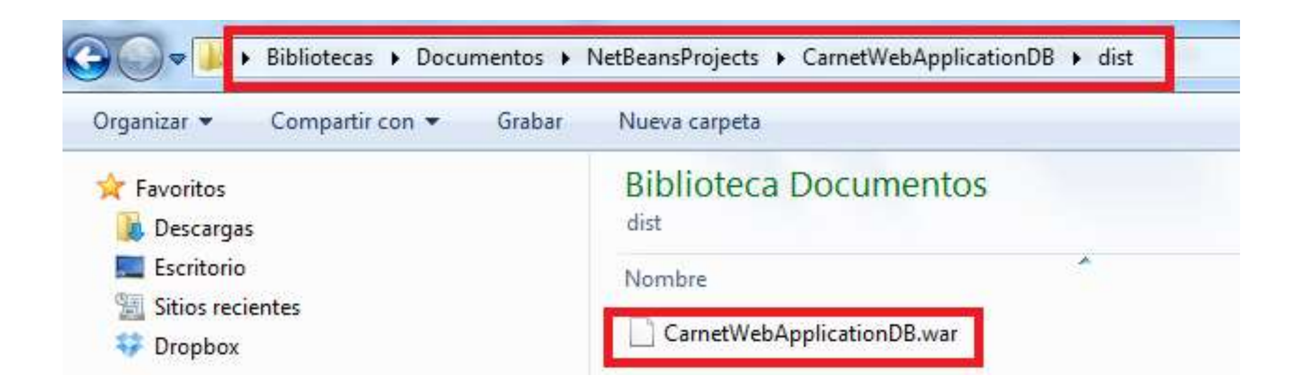
Si tiene demasiados problemas con su proyecto, tome de referencia el que esta subido por la coordinación solo que en las pruebas deberá cambiar el nombre por CarnetWebApplicationBD

6. Desplegando en un entorno de Producción (en casa y aplicable a la prueba final para su proyecto etapa2)

Si queremos desplegar los servicios web en una instalación de GlassFish independiente de NetBeans(descargar e instalar) es necesario compilar y generar el archivo .war, para ello hacemos click derecho sobre el proyecto y seleccionamos "Clean and Build" (es importante que el servidor GlassFish que utiliza NetBeans se encuentre apagado).



Esto nos genera el archivo CarnetWebApplicationDB.war dentro de la carpeta del Proyecto ... \ CarnetWebApplicationDB\dist\



Luego dentro de la carpeta del dominio de GlassFish (C:\glassfish3\glassfish\domains\domain1\lib) es necesario copiar el conector jdbc (mysql-connector-java-5.1.23-bin.jar) para MySQL . (Puede consultar el manual de instalación de Glassfish 3.1.2.2 para Windows).

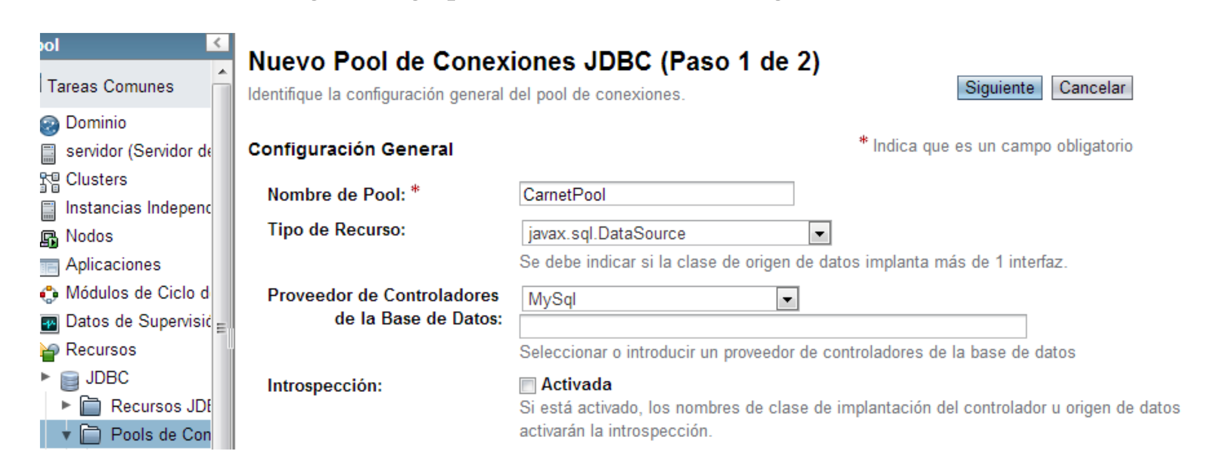
Con esto iniciamos la instancia del dominio de GlassFish y accedemos a la consola de administración en http://localhost:4848



Como primer paso debemos establecer la configuración a la Base de Datos, para ello accedemos la opción "JDBC" → "Pool de Conexiones JDBC" y presionamos el botón "Nuevo"

Página Inicial
Acerca de...
Usuario: admin | Dominio: domain1 | Servidor: localhost
GlassFish™ Server Open Source Edition
Árbol
Tareas Comunes
Dominio
servidor (Servidor de Administrac
Clusters
Instancias Independientes
Nodos
Aplicaciones
Módulos de Ciclo de Vida
Datos de Supervisión
Recursos
JDBC
Recursos JDBC
Pools de Conexiones JDB
Pools de Conexiones JDBC
Para almacenar, organizar y recuperar datos, la mayoría de las aplicaciones utilizan b
través de la API JDBC. Antes de que una aplicación pueda acceder a una base de da
Pools (2)
Nuevo...
Suprimir
Nombre de Pool
Tipo de Recurso
DerbyPool
javax.sql.DataSource
__TimerPool
javax.sql.XADataSource

En la siguiente pantalla introducimos el nombre del pool "CarnetPool" y los valores tal como muestra la imagen, luego presionamos el botón "Siguiente"

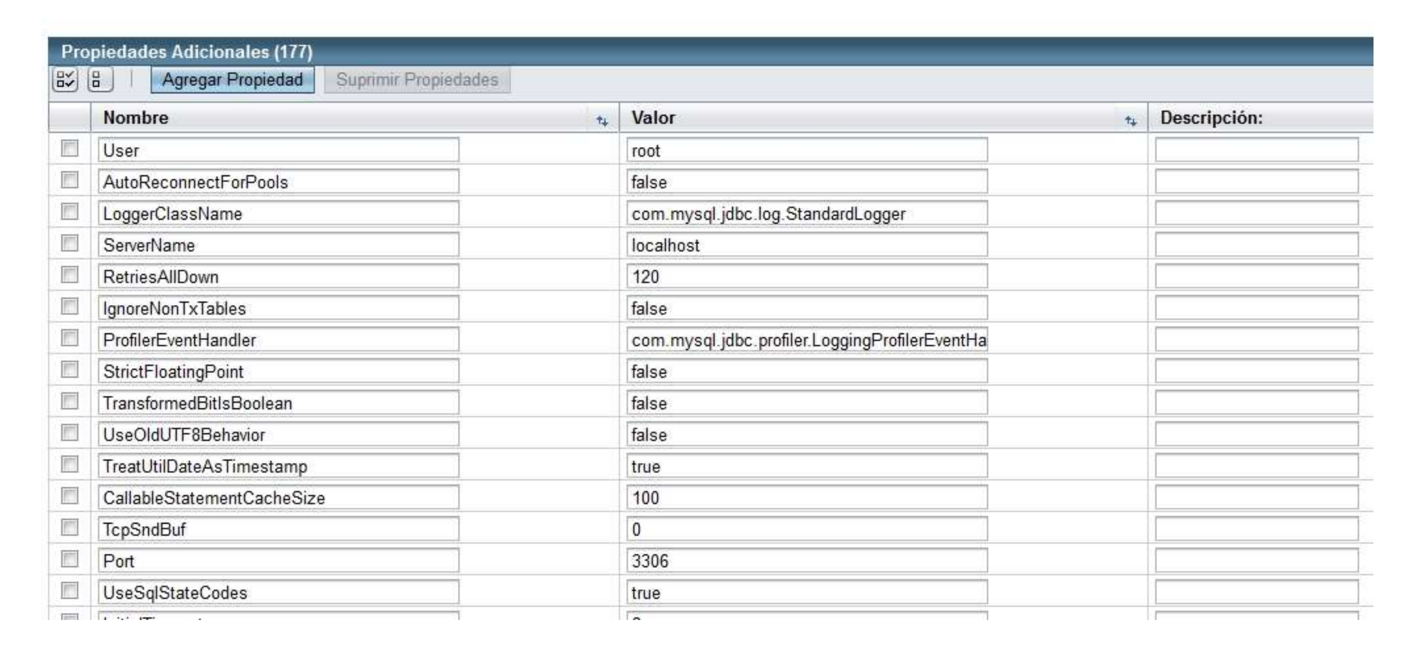




A continuación se muestra la configuración general de la conexión:



En la parte de abajo de esta pantalla se muestran varias propiedades adicionales para la conexión, pero solo se deben buscar y establecer las siguientes:



Propiedades Adicionales (177)

Agregar Propiedad | Suprimir Propiedades

Nombre	Valor	Descripción:
User	root	
AutoReconnectForPools	false	
LoggerClassName	com.mysql.jdbc.log.StandardLogger	
ServerName	localhost	
RetriesAllDown	120	
IgnoreNonTxTables	false	
ProfilerEventHandler	com.mysql.jdbc.profiler.LoggingProfilerEventHa	
StrictFloatingPoint	false	
TransformedBitIsBoolean	false	
UseOldUTF8Behavior	false	
TreatUtilDateAsTimestamp	true	
CallableStatementCacheSize	100	
TcpSndBuf	0	
Port	3306	
UseSqlStateCodes	true	

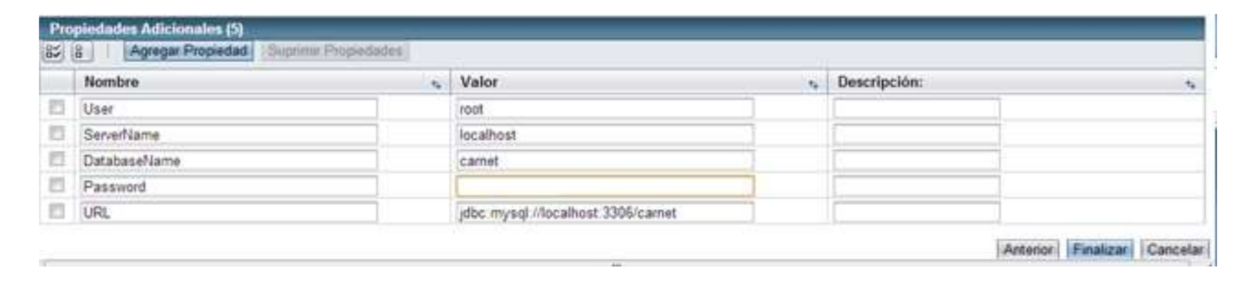
User: root

ServerName: localhost

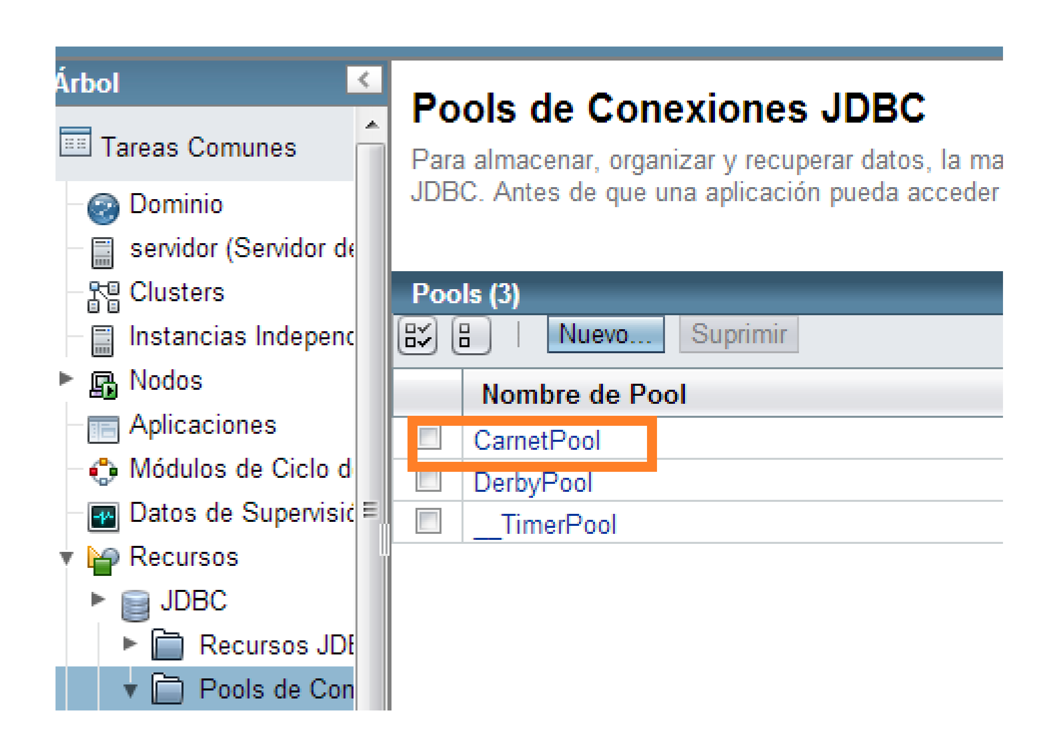
DatabaseName: carnet

Password:

URL: jdbc:mysql://localhost:3306/carnet



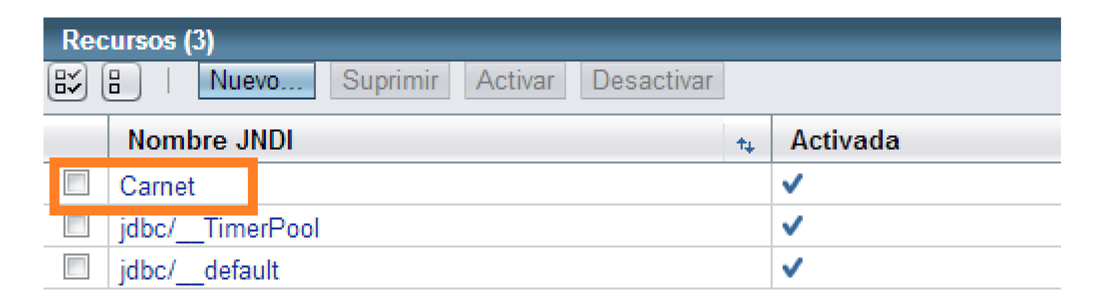
Una vez establecidas las propiedades hacemos click sobre el botón “Finalizar”, nuestra conexión ahora se encuentra creada



Ahora accedemos a la opción “JDBC” → “Recursos JDBC” y hacemos click en “Nuevo”



Es obligatorio que el nombre del recurso sea “Carnet” ya que así fue nombrado dentro del proyecto creado en NetBeans, y le asignamos el pool de conexiones creado anteriormente



Hacemos click sobre el botón “Aceptar” y nuestro recurso será creado:

Como último paso queda desplegar el archivo CarnetWebApplicationDB.war por lo que accedemos a la opción Aplicaciones y hacemos click en el botón “Desplegar”.



Buscamos la ruta del archivo CarnetWebApplicationDB.war y presionamos el botón “Aceptar”



Nuestros servicios están listos ahora dentro del servidor, y podemos acceder mediante las URL's que se probaron de igual manera utilizando el proyecto de NetBeans o si es un servidor externo utilizando la dirección ip correspondiente en lugar de localhost.


Aplicaciones Desplegadas (1)

Desplegar... | Anular Despliegue | Activar | Desactivar | Filtro:

	Nombre	Activada	Motores	Acción
☐	CarnetWebApplicationDB	✔	ejb, web	Iniciar | Volver a Desplegar | Volver a Cargar

Usando el addon de Mozilla, haz las siguientes pruebas

Pruebas de casa

Guardar tus datos materia en servidor

Método: POST

URL:
http://IPSERVER:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/

Pruebas en Servidor de EISI

Guardar tus datos materia en servidor

Método: POST

URL:
http://172.16.14.14:8080/CarnetWebApplicationDB/webresources/sv.ues.fia.carnet.entidad.alumno/

Casos especiales…

Si quieres que tu entorno de desarrollo(PC1, PC2, etc) pueda conectarse con el de producción(Servidor) deberás ir a modificar dos cosas al servidor.

El archivo my.cnf

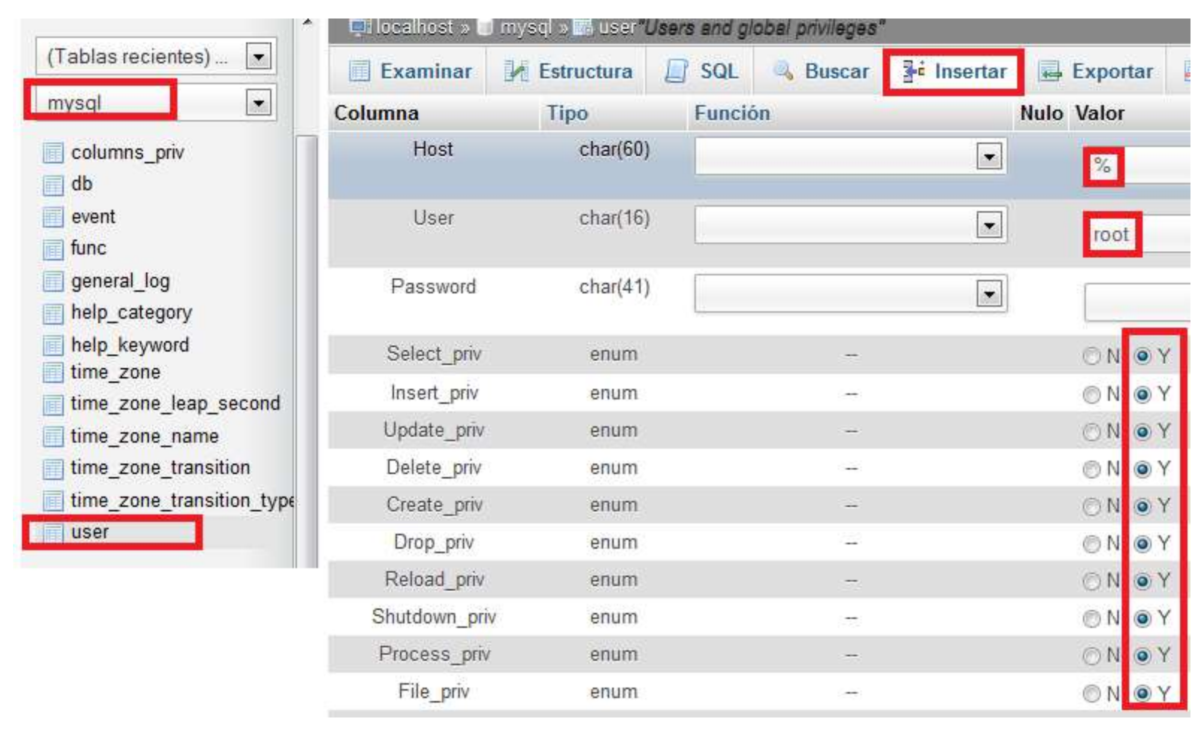
```
Modificamos la linea…
bind-address        = 127.0.0.1
Por…
bind-address        = 0.0.0.0
Reiniciamos el servidor
```

Si no existe…. Se crea la línea.



Añadir un usuario genérico a mysql.

Presione clic en base mysql, tabla user, luego insertar.



El porcentaje es un comodín para que la conexión tome el nombre de host según la ip del dhcp y no de forma estática como en otros gestores de base como Oracle.